CLP

CP2301

e-book

SUMÁRIO

I. APRESENTAÇÃO. 03
II. INTRODUÇÃO. 04
III. CONCEITOS FUNDAMENTAIS. 06
IV. DISPOSITIVOS DE ENTRADA E SAÍDA. 16
V. ASPECTOS DO HARDWARE – SIMATIC S7-200. 19
VI. ASPECTOS DOSOFTWARE – STEP 7 – MICRO/WIN. 27
VII. ESTRUTURA DO PROGRAMA STEP 7 – MICRO/WIN. 41
VIII. PROJETANDO NO S7-200. 50
IX. PASTA DE INSTRUÇÕES. 55
X. MÓDULOS DE EXPANSÃO ANALÓGICOS. 93
XI. DISPLAY DE TEXTO – TD200. 113
XII. CONCLUINDO. 114

I. APRESENTAÇÃO

O tema Automação Industrial, tratado nesta apostila, é hoje um importante campo de atuação, para o qual convergem significativos avanços científicos e tecnológicos.

Olhando ao nosso redor podemos identificar, sem dificuldade, as inúmeras aplicações da automação nos dias de hoje: os portões eletrônicos, os sensores de presença, os comandos à distância de equipamentos residenciais, sem falar no sem fim de possibilidades da automação industrial.

Portanto, ingressar no mundo da automação significa ingressar também numa nova fronteira da tecnologia. Os avanços são grandes e freqüentes, ocorrendo numa velocidade impensável há anos atrás. Você terá, então, um espaço vastíssimo para continuar estudando e aprendendo.

Nos capítulos iniciais apresentamos um pouco da história da automação e conteúdos básicos como: sistemas numéricos aplicáveis e os dispositivos de entrada/saída: sensores, botoeiras, chaves fim de curso, pressostatos, relés térmicos, contatores, bobinas, sinaleiras, dentre outros.

Por fim, abordamos o software STEP7 – Micro/Win, seus ambientes de navegação, componentes e suas principais aplicações na indústria.

Leia os textos com muita atenção, procure respostas para as questões e exercícios que lhe serão colocados, reflita sobre eles, pesquise, indague; enfim aproveite todas as oportunidades para saber mais.

Bons estudos!

II. INTRODUÇÃO

Durante a década de 50, os dispositivos eletromecânicos foram os recursos mais utilizados para efetuar controles lógicos e de intertravamentos nas linhas de produção e em máquinas isoladas. Tais dispositivos, baseados principalmente em relés, tinham especial importância na indústria automobilística em que a complexidade dos processos produtivos envolvidos exigia, frequentemente, instalações em painéis e cabines de controle com centenas de relés e, conseqüentemente, um número maior ainda de interconexões deles.

Tais sistemas de controle, apesar de funcionais, apresentavam problemas de ordem prática bastante relevante. Como as instalações possuíam uma grande quantidade de elementos, a ocorrência de uma falha qualquer significava o comprometimento de várias horas, ou mesmo dias de trabalho de pesquisa e correção do elemento faltoso. Além disto, pelo fato de os relés apresentarem dimensão física elevada, os painéis ocupavam grande espaço, o qual deveria ser protegido contra umidade, aquecimento, gases inflamáveis, oxidação, poeira, etc.

Outro fator ainda comprometedor das instalações a relés era o fato de que, como a programação lógica do mesma exigiam interrupções no processo produtivo, a fim de se reconectarem os elementos. Interrupções industrial.

Saiba Mais...

1.HARDWIRED: embutido em um sistema através de um componente de **Hardware**, como circuitos lógicos, em vez de ser implementado através de programação.
Fisicamente conectado a um sistema ou **Rede**, através, por exemplo, de uma placa de conexão de **Rede** e um cabo.
Fonte: http://xoopscube.com.br/modules/xclDic/entry.php?entryID=5647 acesso em 17/11/2008

Como conseqüência, tornava-se obrigatória a atualização das listas de fiação como garantia de manter a documentação do sistema.

Com o advento da tecnologia de estado sólido, desenvolvida, a princípio, em substituição às válvulas a vácuo, alguns dispositivos transistorizados foram utilizados no final da década de 50 e início dos anos 60, sendo que tais dispositivos reduziam muitos dos problemas existentes nos relés. Porém, foi com o surgimento dos componentes eletrônicos integrados em larga escala, que novas fronteiras se abriram ao mundo dos computadores digitais e, em especial, às tecnologias para a automação industrial.

Assim, a primeira experiência de um controle de lógica que permitisse a programação por recursos de software foi realizada em 1968, na divisão de hidramáticos da GM (General Motors). Aliado ao uso de dispositivos periféricos, capazes de realizar operações de entrada e saída, um minicomputador com sua capacidade de programação pode obter vantagens técnicas de controle que suplantaram o custo que tal implementação representou na época. Iniciava-se a era dos controladores de lógica programável.

Essa primeira geração de CLP, como poderia ser denominada, recebeu sensíveis melhorias com o advento dos microprocessadores ocorrido durante os anos 70. Assim, não se tornava necessário o uso de computadores de grande porte, tornando-o uma unidade isolada. Foram adicionados ainda recursos importantes, tais como interfaces de operação e programação facilitadas ao usuário, instruções aritméticas e de manipulação de dados poderosas, recursos de comunicação por meio de redes de CLP, possibilidades de configuração específica a cada finalidade, por meio de módulos intercambiáveis, dentre outras inúmeras vantagens encontradas nos modelos comerciais que estão atualmente disponíveis.

No Brasil, porém, é na década de 80, que o CLP veio a proliferar na indústria, primeiramente pela absorção de tecnologias utilizadas nas matrizes das multinacionais. Atualmente, com a crescente redução no custo do CLP, observa-se o incremento de sua utilização nas indústrias em geral, independente de seu porte ou ramo de atividades.

III. CONCEITOS FUNDAMENTAIS

Nesta seção, discutiremos rapidamente conceitos básicos e importantes para a compreensão do funcionamento do CLP e de sua programação.

Iniciamos com a diferenciação entre automação e automatização, analisaremos os sistemas de numeração usados no nosso cotidiano e aqueles utilizados em dispositivos eletrônicos, passaremos pelas portas lógicas e finalizaremos com os tipos de memória encontrados nos PLC.

· Automatização e Automação

> **Saiba Mais...**
>
> **2 - Controle em malha aberta** consiste em aplicar um sinal de controle na entrada de um sistema, esperando-se que, na saída, a variável controlada consiga atingir um determinado valor ou apresente um determinado comportamento desejado.
> http://pt.wikipedia.org/wiki/Controlador_de_malha_aberta

O termo automatização se difundiu desde a construção das primeiras máquinas e se consolidou com a revolução industrial, portanto, a automatização está indissoluvelmente ligada à sugestão de movimento automático, repetitivo, mecânico e é sinônimo de mecanização, reproduzindo ação. Caso se entenda que tal mecanização implica somente ação cega, sem correção, tem-se um sistema no qual a saída independe da entrada, ou seja, não existe uma relação entre o valor desejado para um sistema e o valor recebido por este, por meio da variável responsável por sua atuação.

Diz-se que esse tipo de controle se dá por **malha aberta**[2]. Neste caso, o sistema terá sempre o mesmo comportamento esperado, pois ele é determinado por leis físicas associadas ao hardware utilizado. Hardware que pode ser de natureza mecânica, elétrica, térmica, hidráulica, eletrônica ou outra.

A automação é um conceito e um conjunto de técnicas por meio das quais se constroem sistemas ativos, capazes de atuar com uma eficiência ótima pelo uso das informações recebidas do meio sobre o qual atuam.

> **Saiba Mais...**
>
> **3 - Controle em malha fechado:** Também chamado de controle retroativo (realimentação) necessita de informações da saída do controlador, através de elementos sensores ou transdutores, compara o sinal da saída com o *set-point* e corrige a saída, caso a mesma esteja desviando-se dos parâmetros programados.
>
> http://pt.wikipedia.org/wiki/Controle_de_Malha_Fechada

Com base nas informações, o sistema calcula a ação corretiva mais apropriada para a execução da ação. Esta é uma característica de sistemas em **malha fechada**[3], conhecidos como sistemas de realimentação, ou seja: aqueles que mantêm uma relação expressa entre o valor da saída em relação ao da entrada de referência do processo. Essa relação entrada / saída serve para corrigir eventuais valores na saída que estejam fora dos valores

desejados.

Na automação, prevê-se o uso extensivo dos mesmos conceitos associados à automatização. Entretanto, o nível de flexibilidade atribuído ao sistema é bem mais elevado pelo fato de estar associado ao conceito de software. Tal recurso provê, a um sistema dotado de automação, a possibilidade de ser alterado radicalmente todo o comportamento automatizado, a fim de, intencionalmente, produzir–se uma gama diferenciada de resultados.

A automação industrial se verifica sempre que novas técnicas de controle são introduzidas num processo. Pode-se dizer que automação industrial é oferecer e gerenciar soluções, pois ela sai do nível operacional do chão de fábrica para voltar seu foco para o gerenciamento da informação.

- **Sistemas de Numeração**

O modo como contamos as quantidades vem do fato de possuirmos 10 dedos. Assim, tomando os dedos das mãos podemos contar objetos com facilidade até certo ponto. O ponto crítico ocorre quando temos quantidades maiores do que 10. O homem resolveu o problema passando a indicar também a quantidade de mãos ou de vezes em que os dez dedos eram usados.

Assim, quando dizemos que temos 35 objetos, o 3 indica que temos “três mãos cheias” ou três dezenas mais 5 objetos. O 3 tem peso 10.

O que aconteceria se tivéssemos um número diferente de dedos, por exemplo, 3 em cada mão? Isso significaria, em primeiro lugar, que em nosso sistema de base 6 (e não base 10) só existiriam 6 algarismos para representar os números: 0,1, 2, 3, 4 e 5.

Para representar uma quantidade maior do que 6 teríamos de usar mais de um algarismo. Assim, para indicar 9 objetos na base 6, teríamos “uma mão cheia com 6” e mais 3. Isso daria 13. Veja, então, que no número “13” na base 6, o número 1 tem peso 6, enquanto que o número 3 tem o seu valor normal.

Analisando os sistemas numéricos, o sistema que usamos no nosso cotidiano – sistema numérico decimal – não é prático para operações matemáticas dos computadores porque é necessário usar 9 algarismos diferentes para efetuar as contagens.

No computador é fácil usar 0 e 1 porque podemos representar 0 com ausência de tensão elétrica e representar 1 com a presença de tensão elétrica. Por isso, são usados outros sistemas numéricos: o binário e o hexadecimal.

Estudaremos rapidamente como são esses sistemas numéricos e como representar

um mesmo número nesses três sistemas numéricos.

- **Sistema Numérico Decimal**

Este é o sistema onde são utilizados dez algarismos (0, 1, 2, 3, 4, 5, 6, 7, 8 e 9) e a sua posição na grafia do número dá a ele um valor diferente. Deste modo, no número 529, o número 5 vale 5 centenas, o 2 são 2 dezenas e o 9 representa 9 unidades.

- **Sistema Numérico Binário**

Este sistema utiliza apenas dois algarismos (0 e 1), daí o nome binário. A posição dos algarismos na escrita do número dará a eles um valor diferente.

O uso deste sistema é amplamente difundido na eletrônica porque permite a associação com a presença de tensão (corresponde ao algarismo 1) ou ausência de tensão (corresponde ao algarismo 0).

É o sistema utilizado nas operações matemáticas dos computadores porque eles trabalham com dois níveis de tensão: a presença de tensão (ligado) representa o número 1, enquanto que a ausência da mesma representa o número 0.
Quando dizemos que o agrupamento possui 1 byte composto por 8 bits, estamos dizendo que o número é formado por 8 algarismos composto por 0 e 1 e cada algarismo é 1 byte.

O número formado por 4 bits é chamado nibble e o número formado por 16 bits (ou 2 bytes) é chamado word.

Na matemática, operações com **zero** e **um** no sistema binário são realizadas com a álgebra booleana. Os números do computador são escritos com oito dígitos (bits) formados por zeros e uns. A posição do dígito atribuirá a ele um valor diferente, assim como ocorreu com o sistema binário.

Conversão de binário para decimal

Para converter um número binário para o número decimal equivalente basta multiplicar cada dígito pela potência de 2 relativa à posição por ele ocupada e somar os resultados. Assim, por exemplo, o número binário 101 equivale ao número 5 no sistema decimal.

$$101 = 1x22 + 0x21 + 1x20 = 1x4 + 0x2 + 1x1 = 4 + 0 + 1 = 5$$

Da mesma forma como acontece no sistema decimal, os números fracionários são expressos em potências de expoente negativo. Assim, por exemplo, o número binário 0,01 equivale ao número 0,25 no sistema decimal.

0,01 = (0x2-1) + (1x2-2) = (0x1÷21) + (1x1÷22) = (0x1÷2) + (1x1÷4) = (1÷4) = 0,25

Vamos praticar? Objetivo: exercitar a conversão de decimal para binário.

Exercício - Converta para o sistema decimal os seguintes números binários.

a) 1011101.....

b) 0,1101 .....

c) 11001,00101.....

Lembre-se !

Durante a leitura desta apostila: as soluções correspondentes aos exercícios serão tratadas pelo docente no desenvolvimento das aulas.

- **Conversão de decimal para binário**

Dividir o número decimal sucessivamente por dois até obter zero. Os restos de cada operação formam o novo número binário, sendo o valor do primeiro resto, o dígito menos significativo, e o último, o mais significativo.

Vamos praticar? Objetivo: exercitar a conversão de binário para decimal.

Exercício - Converta para o sistema binário os seguintes números decimais.

a) 66.....

b) 227 .....

Para converter a parte fracionária de um número decimal para binário, multiplica-se sucessivamente a parte fracionária por 2. A parte inteira do resultado de cada multiplicação é um dígito binário do novo número, sendo o valor da primeira multiplicação o dígito mais significativo e, o último, o menos significativo.

O critério de parada depende do número de dígitos significativos que pretendemos no resultado.

Vamos praticar? Objetivo: exercitar a conversão de decimal para binário

Exercício - Converta para o sistema binário os seguintes números

a) 0,625.....
b) 0,32.....

- **Sistema Numérico Hexadecimal**

O sistema hexadecimal é um sistema de numeração posicional que representa os números em base 16, portanto empregando 16 algarismos: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E e F. Devido ao sistema decimal, geralmente usado para a numeração, apenas dispor de dez símbolos, deve-se incluir seis letras adicionais para completar o sistema hexadecimal. Deve-se notar que A16 = 1010, B16 = 1110 e, assim, sucessivamente.

Como o sistema binário possui apenas dois algarismos, para se representar um número decimal de dois digitos neste sistema devemos utilizar pelo menos 4 digitos ou 4 bits (1010 = 10102). Os computadores evoluíram rapidamente para sistemas de processamento baseados em 16 bits, daí porque o sistema hexadecimal passou a ser muito utilizado, pois ele gera representações numéricas compactas e as conversões entre hexadecimal e binário são simples.

Conversão de hexadecimal para a base decimal

Para converter um número hexadecimal para o número decimal equivalente basta multiplicar cada dígito pela potência de 16 relativa à posição por ele ocupada e somar os resultados.

Vamos praticar? Objetivo: exercitar a conversão de hexadecimal para decimal.

Exercício - Converta para o sistema decimal os seguintes números hexadecimais.

a) 10, 1A
b) 13C.....

Para converter números decimais em hexadecimais usa-se um método semelhante ao que se utiliza para converter números decimais em binários. O método consiste em dividir o número decimal sucessivamente por dezesseis até obter zero. Os restos de cada operação formam o novo número hexadecimal, sendo o valor do primeiro resto o dígito menos significativo, e o último, o mais significativo.

Vamos praticar? Objetivo: exercitar a conversão de decimal para hexadecimal.

Exercício - Converta para o sistema hexadecimal os seguintes números decimais:
a) 327.....
b) 418.....

Para converter a parte fracionária de um número hexadecimal usa-se também um método semelhante ao usado para converter números fracionários decimais para o sistema binário. Neste caso, multiplica-se sucessivamente a parte fracionária por 16. A parte inteira do resultado de cada multiplicação é um dígito hexadecimal do novo número, sendo o valor da primeira multiplicação o dígito mais significativo e, o último, o menos significativo. O critério de parada depende do número de dígitos significativos que pretendemos no resultado.

Vamos praticar? Objetivo: exercitar a conversão de decimal fracionário para hexadecimal.

Exercício - Converta para o sistema hexadecimal os seguintes números decimais:
a) 0,625
b) 0,32 . .....

· **Conversão de hexadecimal para a base binária**

As conversões da base 16 para a base binária são extremamente simples, daí ser o sistema hexadecimal o escolhido mais frequentemente para substituir o sistema binário.

Cada dígito hexadecimal equivale a quatro dígitos binários. Para converter um número binário para o sistema hexadecimal agrupam-se os dígitos binários em grupos de quatro, da direita para a esquerda, e substitui-se cada um destes grupos por um dígito hexadecimal de valor equivalente.

Vamos praticar? Objetivo: exercitar a conversão de binário para hexadecimal.

Exercício - Converta para o sistema hexadecimal os seguintes números binários:

a) 11010110.....
b) 10011111,101.....

Para converter um número binário para o sistema hexadecimal o método é o inverso deste, ou seja, substitui-se cada dígito hexadecimal por um grupo de 4

dígitos binários com valor equivalente.

Resumindo, na tabela 01, temos a relação de equivalência entre os números decimais, binários e hexadecimais de 0 a 15.

Tabela 01 - Equivalência entre números decimais, binários e hexadecimais

DECIMAL	BINÁRIO	HEXADECIMAL
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

· Portas Lógicas

Portas lógicas são componentes eletrônicos que recebem um ou mais sinais lógicos de entrada e produzem um sinal lógico de saída.

Sinal lógico de entrada é um pulso elétrico ligado na entrada da porta lógica. A presença de tensão +Vcc é entendida como 1 (verdadeiro) e a sua ausência é interpretada como 0 (falsa).

Sinal lógico de saída é a resposta das operações internas da porta lógica de acordo com a Lógica Matemática ou Lógica de Boole, daí o nome porta lógica. Novamente, a presença de tensão +Vcc na saída é 1 (verdadeiro) e a sua ausência é 0 (falso).

As combinações da entrada e a respectiva resposta na saída são apresentadas em uma tabela denominada Tabela da Verdade.

Veja abaixo o resumo do funcionamento das portas lógicas.

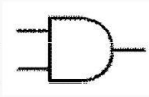
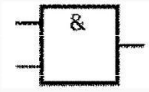
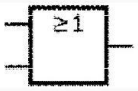
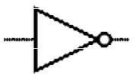
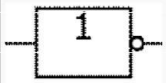
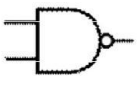
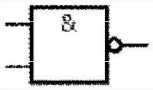
Tipo	Símbolo (Norma ANSI)	Símbolo (Norma IEC)	Função boolean a	Tabela verdade
AND			$A \cdot B$	ENTRADA / SAÍDA A | B | A AND B 0 | 0 | 0 0 | 1 | 0 1 | 0 | 0 1 | 1 | 1
OR			$A + B$	ENTRADA / SAÍDA A | B | A OR B 0 | 0 | 0 0 | 1 | 1 1 | 0 | 1 1 | 1 | 1
NOT			$\overline{A}$	ENTRADA / SAÍDA A | NOT A 0 | 1 1 | 0
NAND			$\overline{A \cdot B}$	ENTRADA / SAÍDA A | B | A NAND B 0 | 0 | 1 0 | 1 | 1 1 | 0 | 1

Tabela 02 - Funcionamento de portas Lógicas

Tipos de Memória

As Memórias são áreas destinadas ao armazenamento de dados. Podem ser de dois tipos: voláteis e não voláteis.

Memórias de armazenamento voláteis são aquelas nas quais uma perda, mesmo que breve, de alimentação de energia resultará na perda da informação armazenada.

Em contrapartida, memórias de armazenamento não voláteis mantêm sua informação mesmo durante a ausência de alimentação, o que às vezes é denominado memória retentiva.

Na organização do sistema de memória dos CLP, encontraremos o uso de

ambos os tipos, incluindo-se ainda, em alguns equipamentos, um sistema de fornecimento de energia via baterias ou acumuladores, a fim de manter os dados que estão armazenados em memórias voláteis.

Os tipos de memórias e como suas principais características afetam a forma de armazenamento/alteração dos dados são relacionados em seguida:

RAM: (Random Access Memory) MEMÓRIA DE ACESSO ALEATÓRIO memória que permite acesso a qualquer posição em qualquer ordem, sem ter que acessar seqüencialmente a partir do primeiro elemento.

É o tipo de memória volátil mais amplamente utilizado. Sua principal característica reside no fato de que os dados podem ser gravados e alterados facilmente, ficando a critério das necessidades do usuário.

Nos CLP, são utilizadas para formar uma área de armazenamento temporário, como uma espécie de rascunho de informações, tanto de dados como de programas.

- ROM: (Read Only Memory) MEMÓRIA EXCLUSIVA DE LEITURA.

São memórias especialmente projetadas para manter armazenadas informações que, sob hipótese alguma, poderão ser alteradas.

Assim, sua única forma de acesso é para operação de leitura. Devido a essa característica, elas se encaixam na categoria de memórias não voláteis.

Num CLP, elas podem ser encontradas para o armazenamento do programa executivo, por exemplo.

- PROM: (Programmable Read Only Memory) MEMÓRIA PROGRAMÁVEL EXCLUSIVA DE LEITURA memória exclusiva de leitura que pode ser programada pelo usuário (diferentemente da ROM, que é programada pelo fabricante), porém em uma única operação de gravação que, caso mal sucedida, comprometerá permanentemente a sua utilização.

- EPROM: (Erasable Programmable Read Only Memory) MEMÓRIA EXCLUSIVA DE LEITURA PROGRAMÁVEL E APAGÁVEL.

É um tipo especial de PROM que permite ao usuário efetuar alterações nos dados ali contidos.

O processo de apagamento dos dados pré-armazenados é feito pela exposição temporária do chip a uma fonte de luz ultravioleta.

A EPROM pode se constituir em um excelente meio de armazenamento não volátil do programa de controle que o CLP irá executar, após, porém, o mesmo ter sido elaborado e totalmente isento de erros, enquanto armazenado em RAM.

- EEPROM: (Eletrically Erasable Programmable Read Only Memory) MEMÓRIA EXCLUSIVA DE LEITURA, PROGRAMÁVEL E APAGÁVEL ELETRICAMENTE.

São dispositivos de memória que, apesar de não voláteis, oferecem a mesma flexibilidade de reprogramação existente nas RAM.

Atualmente, existem CLP equipados com EEPROM em seu sistema de memória, devido à sensível vantagem advinda do seu uso. Porém as EEPROM apresentam duas limitações:

- o processo de regravação de seus dados, que só pode ser efetuado após a limpeza das células;
- a vida útil, que é limitada pelo número de reprogramações que ela pode receber.

IV. DISPOSITIVOS DE ENTRADA E SAÍDA

Dispositivos de entrada e saída são utilizados para enviar ou receber sinais do CLP, sejam eles discretos (digitais) ou analógicos.

Fig. 01 – Dispositivos utilizados na automação de sistemas - SIEMENS

- **Sensores**

Dispositivos construídos para detectar a presença ou passagem de materiais metálicos ou não metálicos, por proximidade ou aproximação, sem contato físico. Esta detecção pode ser feita por resistência, capacitância ou indutância, de forma mais ou menos proporcional.

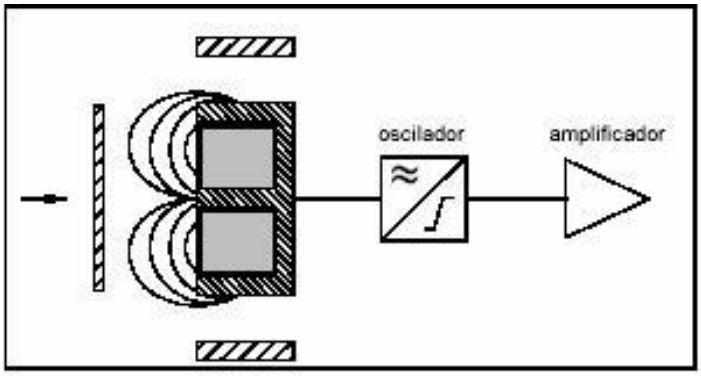


Fig. 02 - Sensor Indutivo de Proximidade

- **Características fundamentais dos sensores para automação**

O sinal de um sensor pode ser caracterizado por:

Linearidade

Grau de proporcionalidade entre o sinal gerado e a grandeza física.

Faixa de Atuação

Intervalo de valores da grandeza em que pode ser usado o sensor.

Histerese

Distância entre os pontos de comutação do sensor.

Sensibilidade

Distância entre a face do sensor e o atuador no instante em que ocorre a

comutação.

Superfície Ativa

Superfície através da qual o campo eletromagnético de alta freqüência se irradia no meio externo.

Fator de Correção

Fator que permite a redução da distância sensora em presença de determinados materiais.

Freqüência de Comutação

Corresponde à quantidade máxima de comutações por segundo. Na tabela a seguir podemos verificar tipos de sensores.

	Família	Tipo	Princípio de funcionamento
Sensores	Indutivos	proximidade	Geração de campo em alta freqüência.
	Capacitivos	proximidade	Geração de campo desenvolvido por oscilador.
	Óticos	difusão	Transmissão e recepção de luz infravermelha que pode ser refletida ou interrompida por um objeto a ser detectado.
		Retro-	
		barreira	
	Ultra-sônicos	difusão	Transmissão ou recepção de onda sonora que pode ser refletida ou
		reflexivo	

Tabela 03 – Tipos de Sensores

- **Botoeiras**

As botoeiras propiciam informações digitais (zero ou um) responsáveis por acionamento e desligamento de motores, válvulas, esteiras, etc.

Fig. 03 – Botoeira Siemens

- **Chaves Fim de Curso**

Os interruptores de posição (ou chaves fim de curso) são dispositivos do tipo chave de impulso, também denominados de “Micro-Switch”, que quando acionados, podem habilitar ou desabilitar qualquer evento do processo.

Fig. 04 – Chave Fim de Curso

Pressostatos

Fig. 05 – Pressostati telemecanique

Os pressostatos têm por função controlar ou regular uma pressão num circuito hidráulico ou pneumático. Eles transformam uma mudança de pressão em sinal elétrico digital, quando a referência fixada for atingida.

V. ASPECTOS DO HARDWARE - SIMATIC S7–200

O CLP Siemens S7-200 possui uma unidade central compacta de processamento (CPU) que reúne:

- A CPU propriamente dita que executa o programa e armazena dados.
- As entradas digitais que monitoram sinais dos equipamentos de campo (tais como sensores e interruptores).
- As saídas digitais que controlam bombas, motores e outros equipamentos dentro do processo.
- A fonte 24Vcc que alimenta a CPU e os módulos de expansão.

IMPORTANTE !
Uma bateria opcional garante a retenção dos dados da memória RAM, na falta de alimentação CA.

A CPU possui leds indicadoros de status que propiciam indicação visual sobre o estado da CPU (RUN, STOP ou SF) e a situação das I/O (entradas e saídas).

SF: Led Vermelho: indica falha no sistema (System Fault).
RUN: Led Verde: a CPU está em ciclo.
Stop: Led Amarelo: o CLP **NÃO** está rodando o programa.

I X.X, entrada genérica. Led verde indica que está energizada.
Q X.X, saída genérica. Led verde indica que está habilitada.

Os módulos de expansão permitem adicionar I/O digitais ou analógicas e são conectadas à CPU, através de um BUS conector (barramento).

SAIBA MAIS....

-CPU 221: não permite expansão.
-CPU 222: 2 módulos de expansão no máximo.
-CPU 224 e CPU 226: no máximo 7 módulos de expansão, sendo destes, no máximo, 2 módulos inteligentes (EM 277 - PROFIBUS-DP).

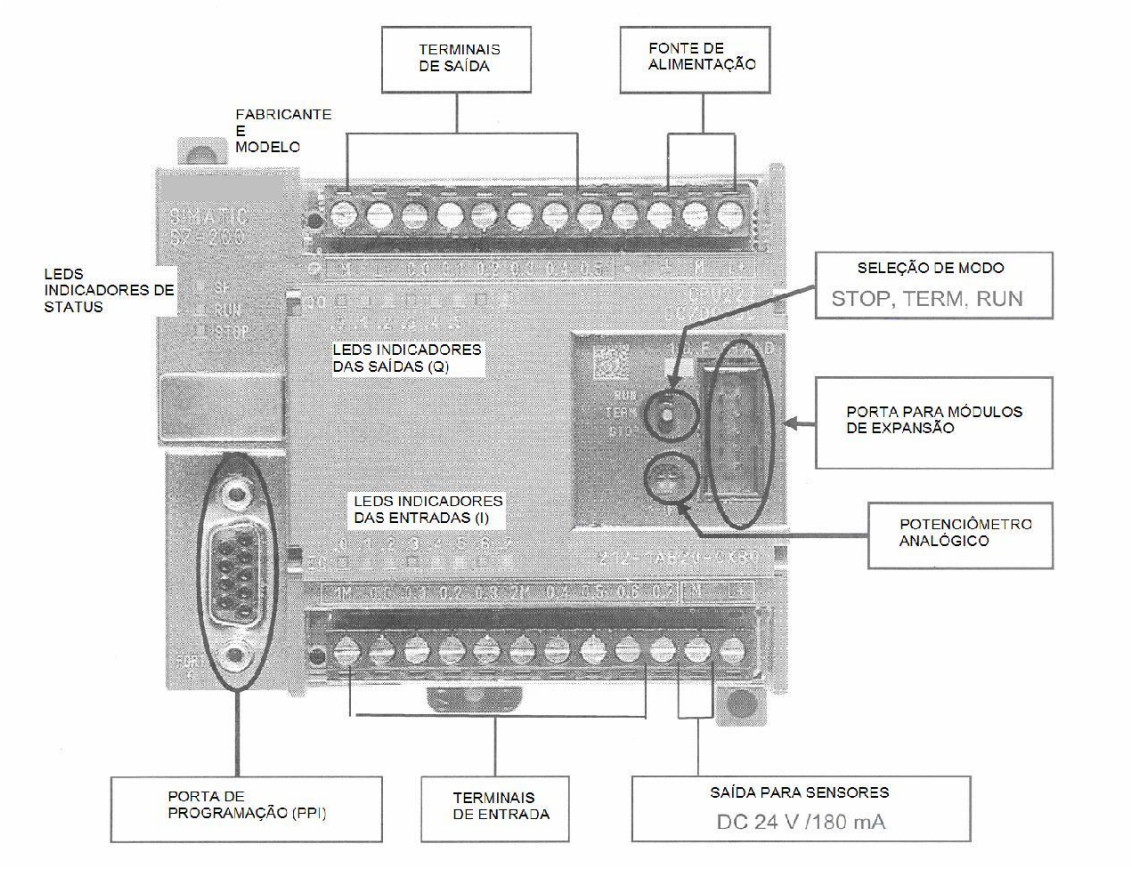


Fig. 06 – Estrutura do CLP S7-200

Alimentação

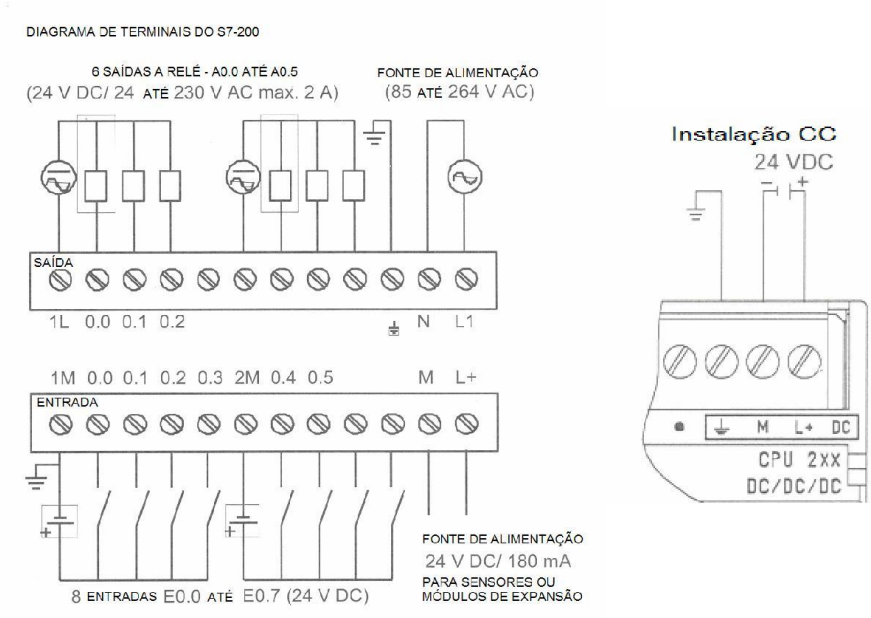


DIAGRAMA DE TERMINAIS DO S7-200
6 SAÍDAS A RELÉ - A0.0 ATÉ A0.5
(24 V DC/ 24 ATÉ 230 V AC max. 2 A)
FONTE DE ALIMENTAÇÃO
(85 ATÉ 264 V AC)
SAÍDA
1L 0.0 0.1 0.2
N L1
1M 0.0 0.1 0.2 0.3 2M 0.4 0.5
M L+
ENTRADA
FONTE DE ALIMENTAÇÃO
24 V DC/ 180 mA
PARA SENSORES OU MÓDULOS DE EXPANSÃO
8 ENTRADAS E0.0 ATÉ E0.7 (24 V DC)
Instalação CC
24 VDC
M L+ DC
CPU 2XX
DC/DC/DC

Fig. 07 - Conexões Elétricas do CLP S7 - 200

DICA !

- Instalar o S7-200 ou equipamento associado a ele, com tensão aplicada, pode causar choque elétrico ou operação defeituosa do equipamento.
- Siga sempre as precauções de segurança apropriadas e assegure-se de que o S7-200 está desenergizado, antes de tentar instalar ou remover qualquer equipamento.

Princípio de Funcionamento

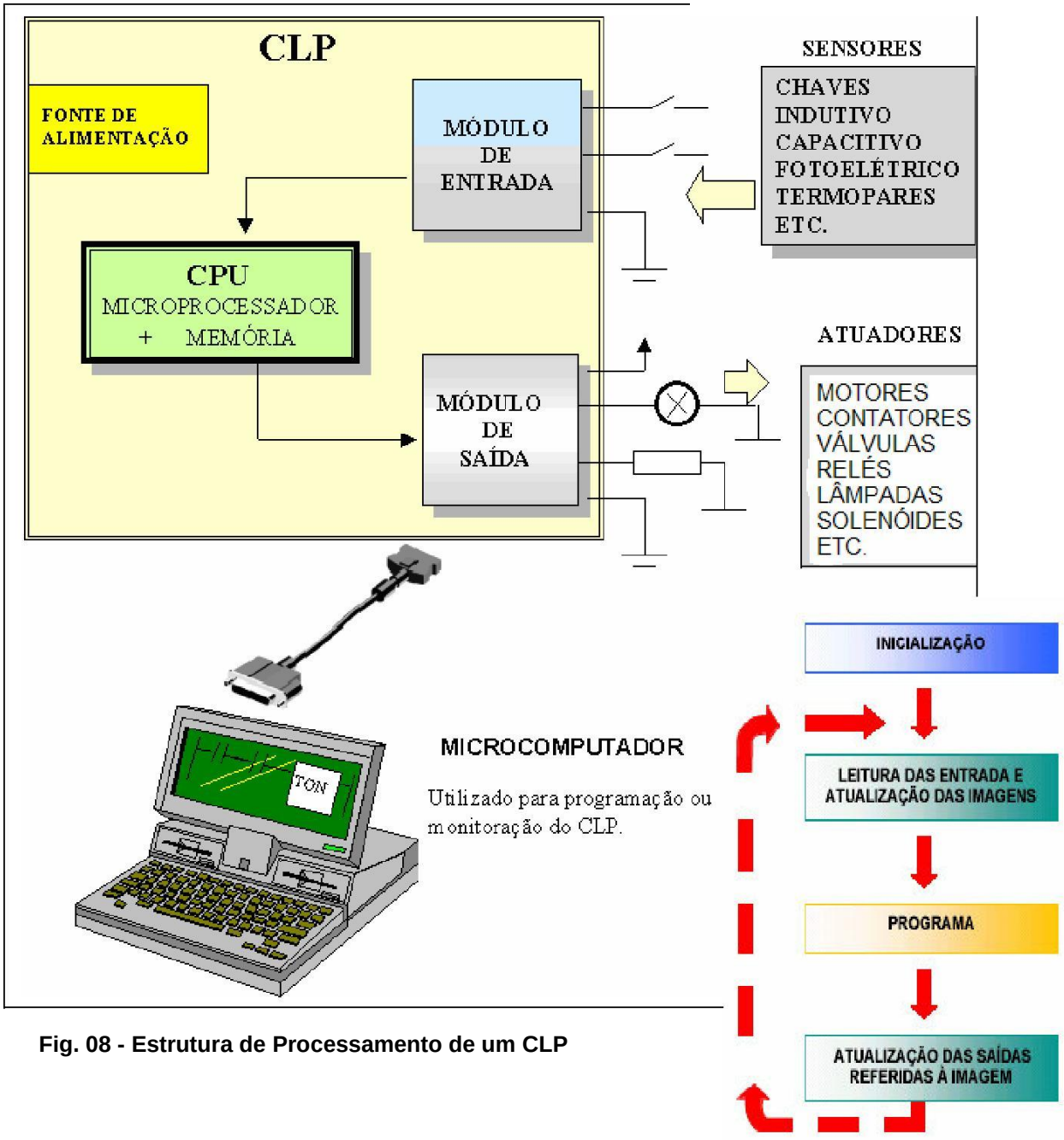


Fig. 08 - Estrutura de Processamento de um CLP

A) Inicialização

No momento em que o CLP é ligado, ele executa uma série de operações pré-programadas, gravadas em seu Programa Monitor.

- Verifica o funcionamento eletrônico da C.P.U., memórias e circuitos auxiliares;
- Verifica a configuração interna e compara com os circuitos instalados;
- Verifica o estado das chaves principais (RUN / STOP , PROG, etc.);
- Desativa todas as saídas;
- Verifica a existência de um programa de usuário;
- Emite um aviso de erro, caso algum dos itens acima falhe.

B) Leitura das entradas e atualização e das imagens

O CLP lê o estados de cada uma das entradas, verificando se alguma foi acionada. Este processo chama-se Ciclo de Varredura ou Scan e normalmente dura microssegundos (scan time).

Após o Ciclo de Varredura, o CLP armazena os resultados obtidos em uma região de memória chamada de “Memória Imagem das Entradas e Saídas”. Ela recebe este nome por ser um espelho do estado das entradas e saídas. Esta memória será consultada pelo CLP no decorrer do processamento do programa do usuário.

C) Programa

O CLP, ao executar o programa do usuário, após consultar a Memória Imagem das Entradas, atualiza o estado da Memória Imagem das Saídas, de acordo com as instruções definidas pelo usuário em seu programa.

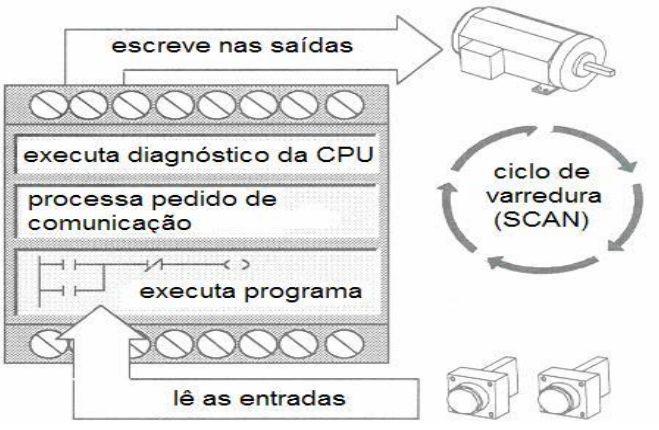


Fig. 09 - Interação entre entradas e saídas de um CLP

D) Atualização das saídas referidas à imagem

O CLP escreve o valor contido na Memória das Saídas, atualizando as interfaces ou módulos de saída. Inicia-se então, um novo ciclo de varredura (etapa B).

- **Modos de Operação da CPU**

O modo de operação da CPU do CLP S7-200 é definido pela chave seletora localizada na própria CPU.

Modo RUN: programa rodando. Não existe possibilidade de transferência de um novo programa, nem a modificação do que está rodan o

Modo STOP: o programa em e e ão é interrompido para que se possa realizar alguma alteração.

Modo TERM: é possível alterar o programa com este rodando, porém, na hora de fazer o download do programa alterado, é necessário levar a CPU para STOP.

- **Protocolos**

Protocolo PPI
(protocolo físico = cabo)

PPI é um protocolo Mestre-Escravo. Neste protocolo, o mestre envia uma ordem e os escravos respondem. Os escravos sempre esperam um comando do mestre. O S7-200 normalmente é um escravo na rede. O limite do protocolo PPI é de 32 mestres em uma rede.

Protocolo MPI
(protocolo físico = cabo)

MPI pode ser um protocolo Mestre-Mestre ou Mestre-Escravo. Se o dispositivo de destino é um CLP S7-300, então a conexão é Mestre-Mestre porque o S7-300 é mestre na rede. Se o dispositivo de destino é um CLP S7-200 CPU, então a conexão será Mestre-Escravo, porque os S7-200 são escravos na rede. Na conexão MPI outro mestre não pode interferir.

Protocolo PROFIBUS
(protocolo lógico = software de gerenciamento de rede)

O protocolo de PROFIBUS é projetado para comunicações de alta velocidade com dispositivos de I/O distribuídos (I/O remoto). Há muitos dispositivos PROFIBUS disponíveis no mercado. Redes PROFIBUS normalmente têm um mestre e vários escravos. O mestre é configurado para saber que tipos de escravos estão na rede e seus endereços. O mestre escreve instruções nos escravos e lê o “feedback” destes.

· **Cabos de Conexão**

Podemos programar o CLP S7-200 utilizando um PC com o software Step7-Micro/Win instalado. A Siemens provê dois meios físicos para conectar o PC ao S7-200.

- Conexão direta usando um cabo conversor PPI (interface ponto a ponto) Multi-Mestre.
- Cartão CP (processador de comunicações) com um cabo conversor MPI (interface multi ponto).

O cabo PPI é o mais comum e econômico método de comunicação entre a porta de comunicação 0 ou 1 do S7-200 e a porta de comunicação serial COM 1 ou COM 2 do PC. Ele também pode ser usado para conectar outros equipamentos de comunicação ao S7-200.

A extremidade do cabo PPI, que se conecta ao PC, é RS-232 e está marcada PC.

A extremidade do cabo PPI, que se conecta ao S7-200, é RS-485 e está marcada PPI.

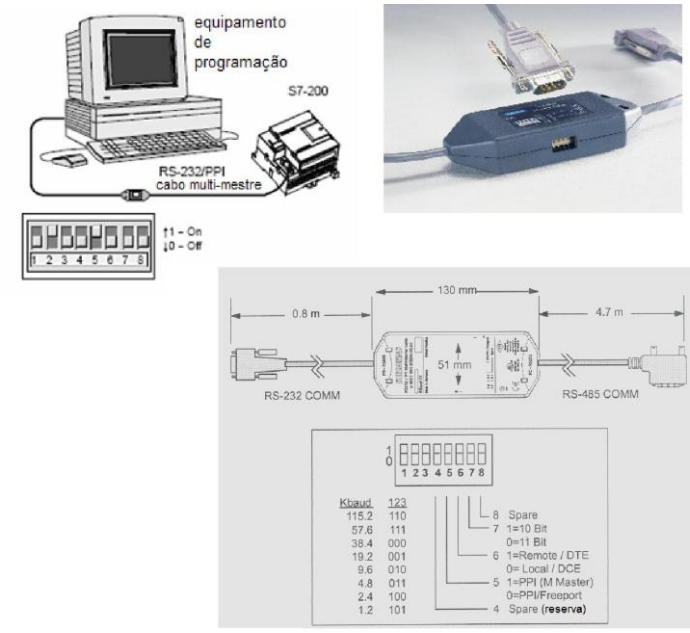


Fig. 10 - Cabo PPI atual (8 chaves)

O cabo RS-232/PPI Multi-Master tem 8 Switches (chaves). Duas delas são usadas para configurar o cabo para operação com o STEP 7 - Micro/WIN.

Se você está conectando o cabo ao PC, selecione PPI mode (chave 5 = 1) e operação **local** (chave 6 = 0).

Se você está conectando o cabo a um modem, selecione PPI mode (chave 5 = 1) e operação **remoto** (chave 6 = 1).

As chaves 1, 2 e 3 selecionam a taxa de transmissão de dados (Baud Rate). O Baude Rate mais comum é 9600, que tem posicionamento de chaves igual a 010.

Escolha o cabo PC/PPI como iterface e selecione a porta RS-232 que você pretende usar no PC. No cabo PPI selecione o endereço da estação e o Baud Rate. Você não precisa fazer outras seleções porque a seleção do protocolo é automática com o cabo RS-232/PPI Multi-Mestre.

Ambos os cabos, USB/PPI e o RS-232/PPI Multi-Mestre, têm LEDs que indicam a atividade de comunicação.

O LED Tx, verde - indica que o cabo está transmitindo informação para o PC. O LED Rx, verde - indica que o cabo está recebendo dados.

O LED PPI, verde - indica que o cabo está transmitindo na network.

- Switches (chaves) 1, 2 e 3 determinam a taxa de transmissão de dados (baud rate).
- Chave 5 seleciona o modo PPI ou PPI/Freeport.
- Chave 6 seleciona modo **local** ou **remoto**.
- Chave 7 seleciona protocolo PPI de 10-bit ou 11-bit.
- Chaves 4 e 8 são spare (reserva).

PG/PC
S7-200 CPU
USB port
SAIBA MAIS....
• O cabo PPI também é fornecido com terminação USB.
• Nesta versão ele é "plug and play" (instalação automática), não há chaves para serem setadas, como na versão anterior.
• Conecte o cabo PPI e o selecione como interface padrão, selecione o protocolo PPI e sete a porta USB no PC.

Fig. 11 - Cabo de Comunicação entre PC e clp

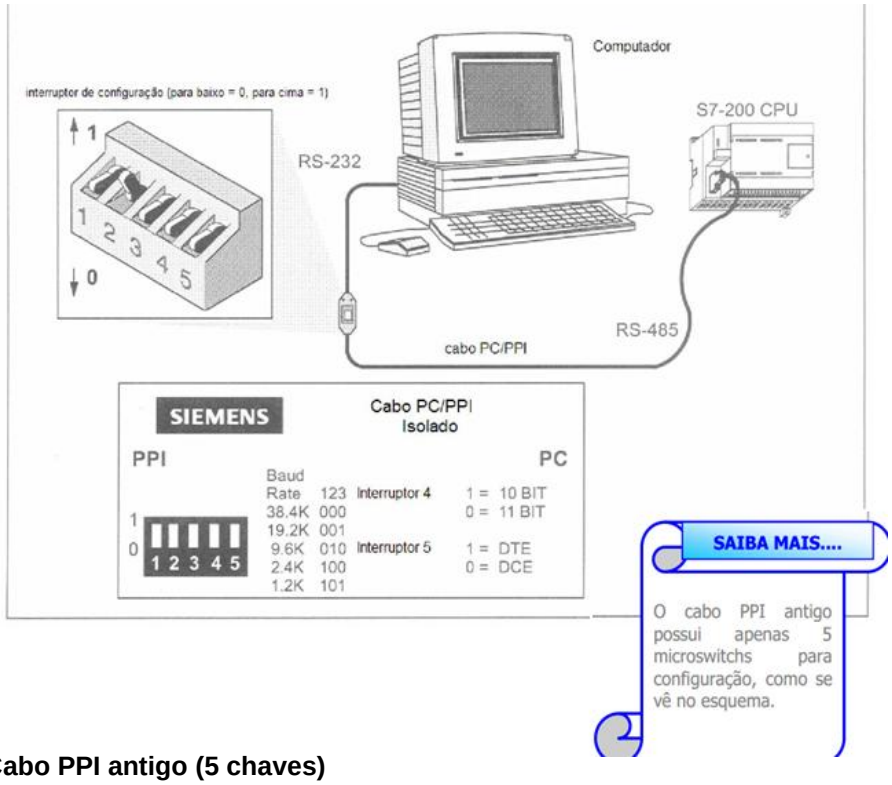


Fig. 12 - Cabo PPI antigo (5 chaves)

VI. ASPECTOS DE SOFTWARE - STEP 7 – MICRO/WIN

O software de programação da linha de equipamentos S7-200 da SIEMENS é o STEP 7 – Micro/Win. Na figura, a seguir, vemos o ambiente de programação.

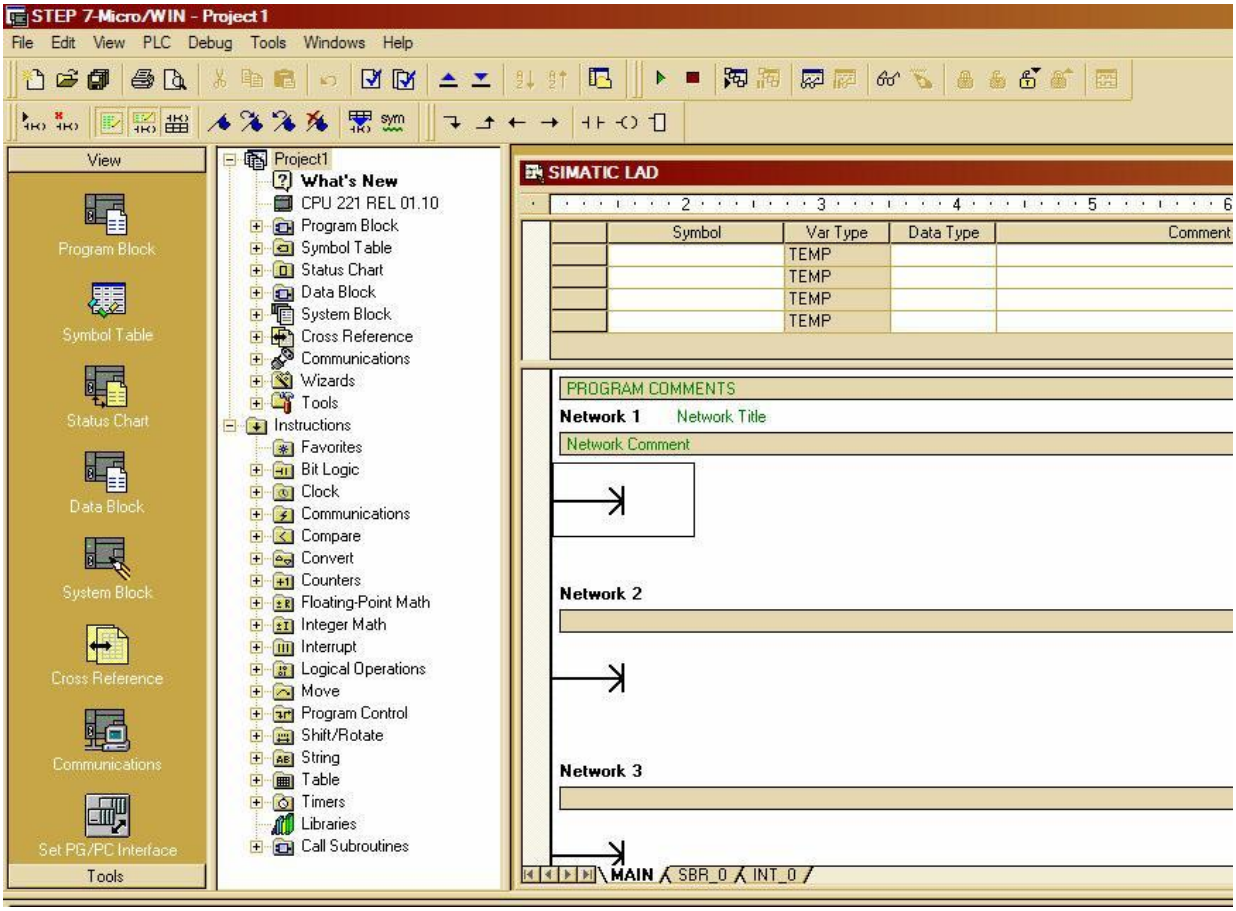


Fig. 13 - Ambiente de Programação STEP 7- Micro/Win

· **Ambiente de Programação**

Barra de Títulos

Onde lemos o nome do software e o nome do projeto.

Barra de Menu (Comandos)

File, Edit, View, PLC, Debug, Tools, Windows e Help

Upload

É utilizado para carregar o programa que está no PC para a memória do CLP.

Download

É utilizado para deslocar o que está na memória do CLP para o PC.

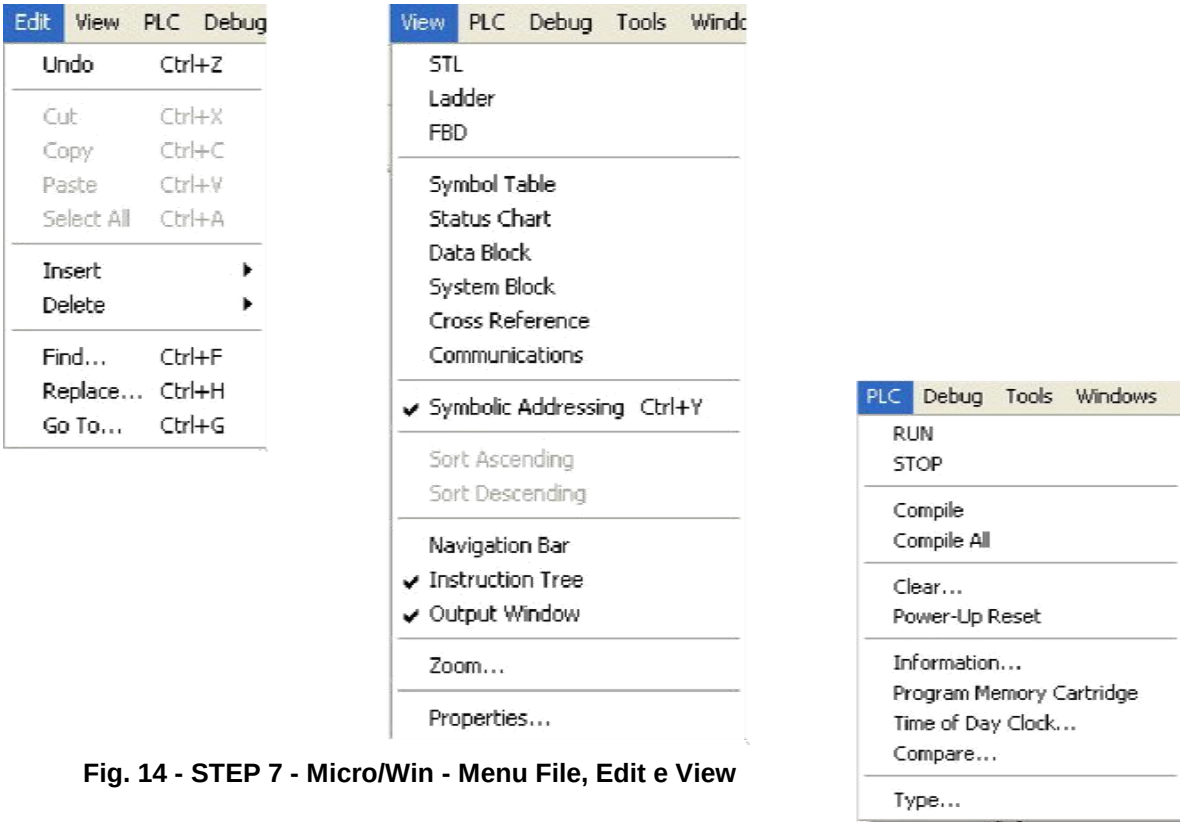


Fig. 14 - STEP 7 - Micro/Win - Menu File, Edit e View

Compile (compilar – converter programa fonte em programa objeto)

É utilizado para compilar o programa. Quando se faz a compilação, o software faz uma varredura no programa em busca de erros.

Clear (limpar)

É utilizado para limpar o programa residente da memória do CLP.

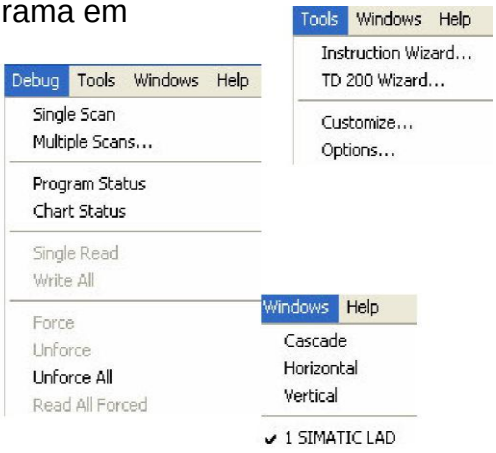
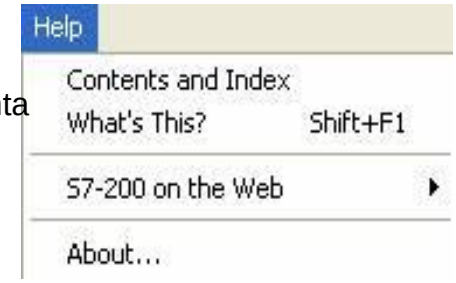


Fig. 15 - STEP 7 - Micro/Win - Menu PLC, Debug, Tools e Windows

Help (ajuda)

Oferece 3 meios para se obter informações:



I. Contents and Index (conteúdo e índice) Apresenta todo o conteúdo por ordem alfabética.

II. What´s This? (O que é isto?)
Ao ser selecionado, aparece ao lado do cursor o símbolo de interrogação (?). Selecionando, com este cursor especial, o item sobre o qual se deseja a informação, abre-se a tela do HELP.

III. S7-200 on the Web

Apresenta alguns sites na Web onde podemos conseguir catálogos, suporte, dicas e outras informações.

Fig. 17 – Menu View

Outros meios produzem o mesmo efeito do recurso **What´s This**?

SAIBA MAIS....

- Selecionar com o mouse o item sobre o qual se deseja informação e clicar "F1".
- Clicar com o botão direito do mouse sobre o item e selecionar HELP.

Dentro do HELP, para filtrar a pesquisa, usar "tópicos da ajuda – índice ou localizar".

Barra de Ferramentas

Onde encontramos as ferramentas usadas para a elaboraç e execução do

programa.

Barra de Status

Parte inferior da tela, onde vemos se estamos em uma tela principal (MAIN) ou em uma sub-rotina (SBR) ou, ainda, em uma rotina de interrupção (INT).

Data Block;

Essas opções também podem ser encontradas em “Instruction Tree” ou pela barra de Menu na opção “View - Component”, como se vê na figura.

I. System Block (bloco de sistema)

No System Block configuramos todas as características da CPU do S7-200.

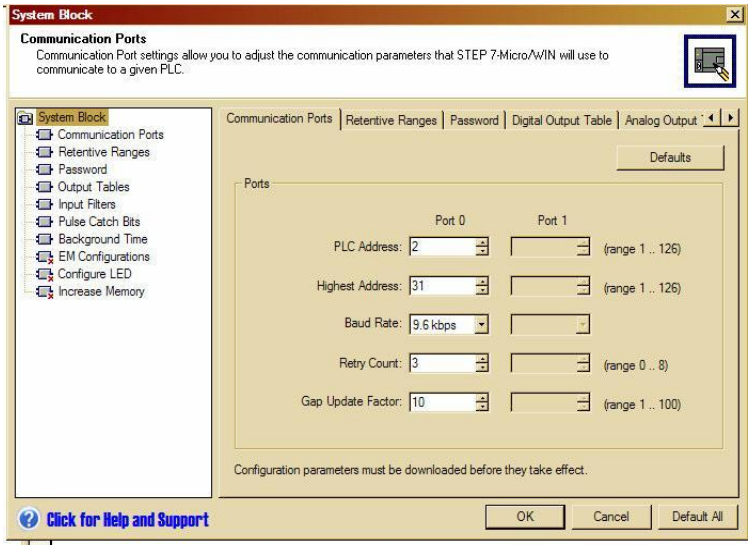


Fig. 18 – Tela do System Block

A) Communication Ports (portas de comunicação)

Nesta pasta configuramos as características de comunicação da CPU. **CLP Address** – Endereço da CPU na rede PPI;

Highest Address – Número máximo de participantes na rede PPI;

· **Baud Rate** – Velocidade de Comunicação (CP – CLP; CLP – CP);

· **Retry Count** – Número de vezes que o sistema tenta se comunicar com o CLP, antes de sinalizar a falha;

· **Gap Update Factor** – Quantos elementos à frente, a CPU deve pesquisar na rede.

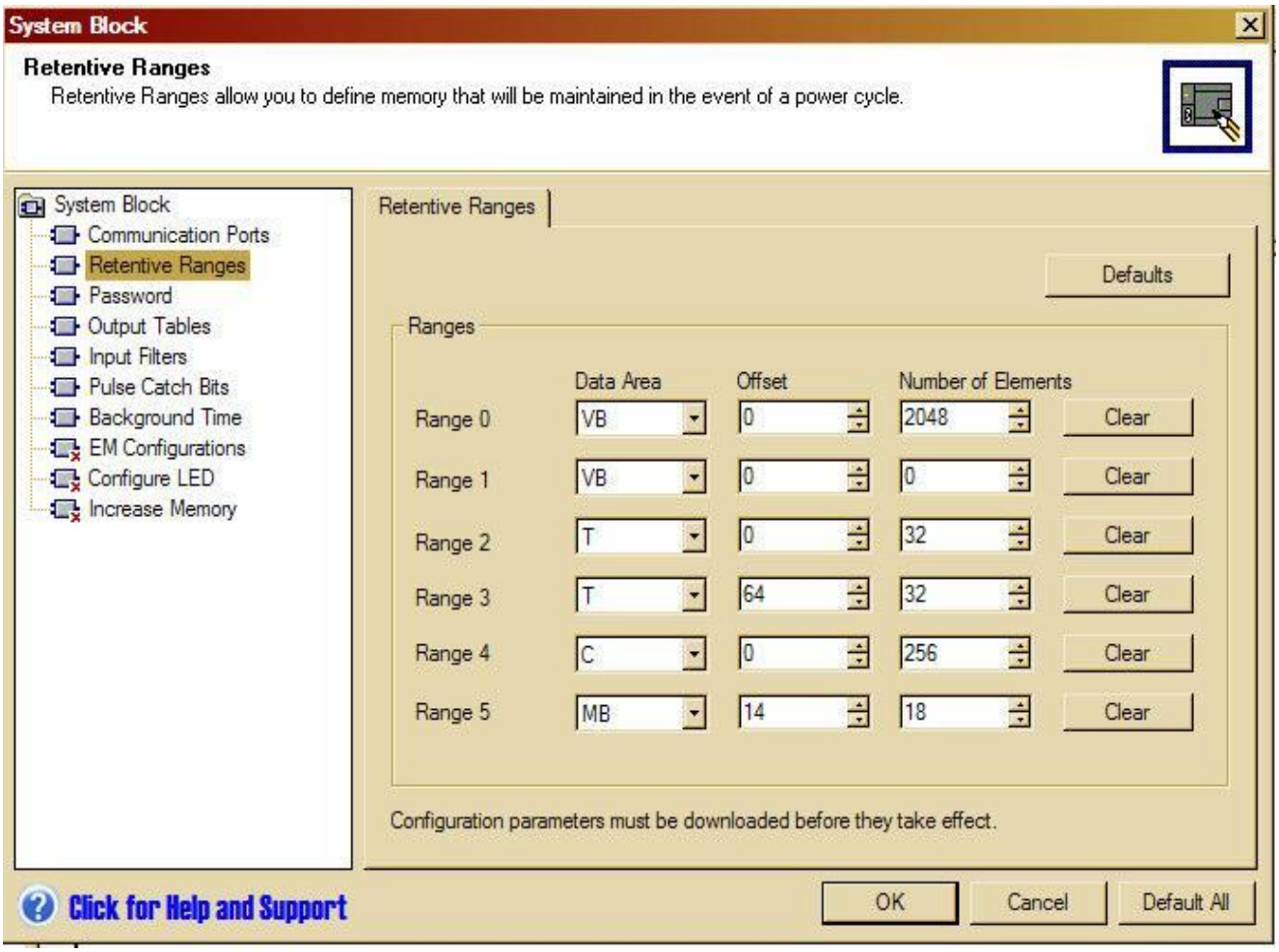


Fig. 19 – Opção Retentive Ranges

B) Retentive Ranges (faixas retentivas)

Nesta pasta configuramos as áreas de memória retentiva (relembrando: memória que não perde a informação, mesmo com a CPU desligada).

· **Data Área -** Estabelece o tipo de memória em cada range.

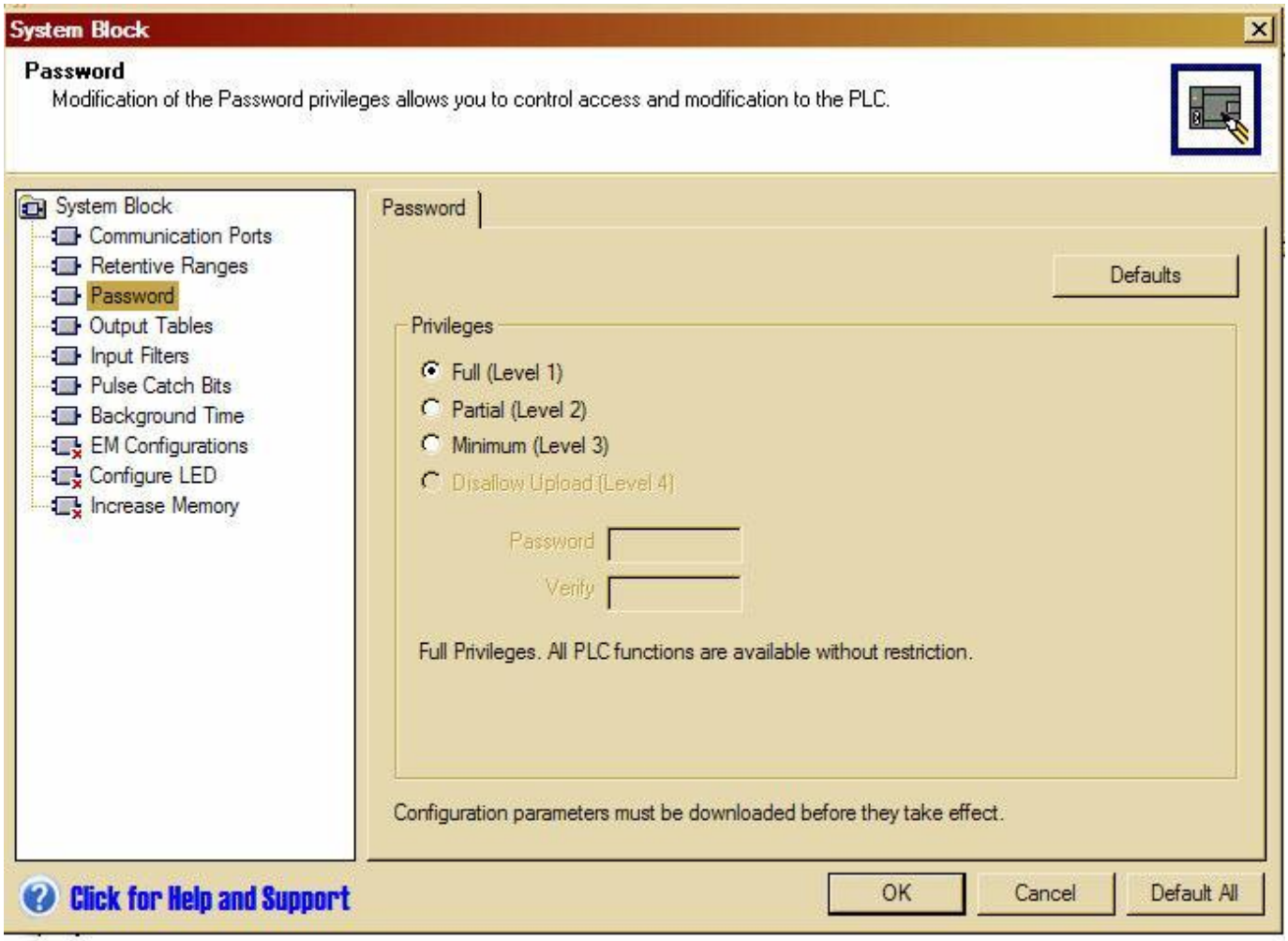
· **Offset -** Endereço inicial da memória.

· **Number of Elements -** Número de elementos que, a partir do endereço inicial, ocupará a área de memória retentiva.

· **Clear -** Botão que limpa os campos.

Defaults - Botão que carrega as caract

Fig. 20 – Opção Password



C) Password (senha)

Nesta pasta podemos inserir uma senha para o acesso parcial ou total da aplicação que está sendo realizada. O tipo de acesso pode ser selecionado:

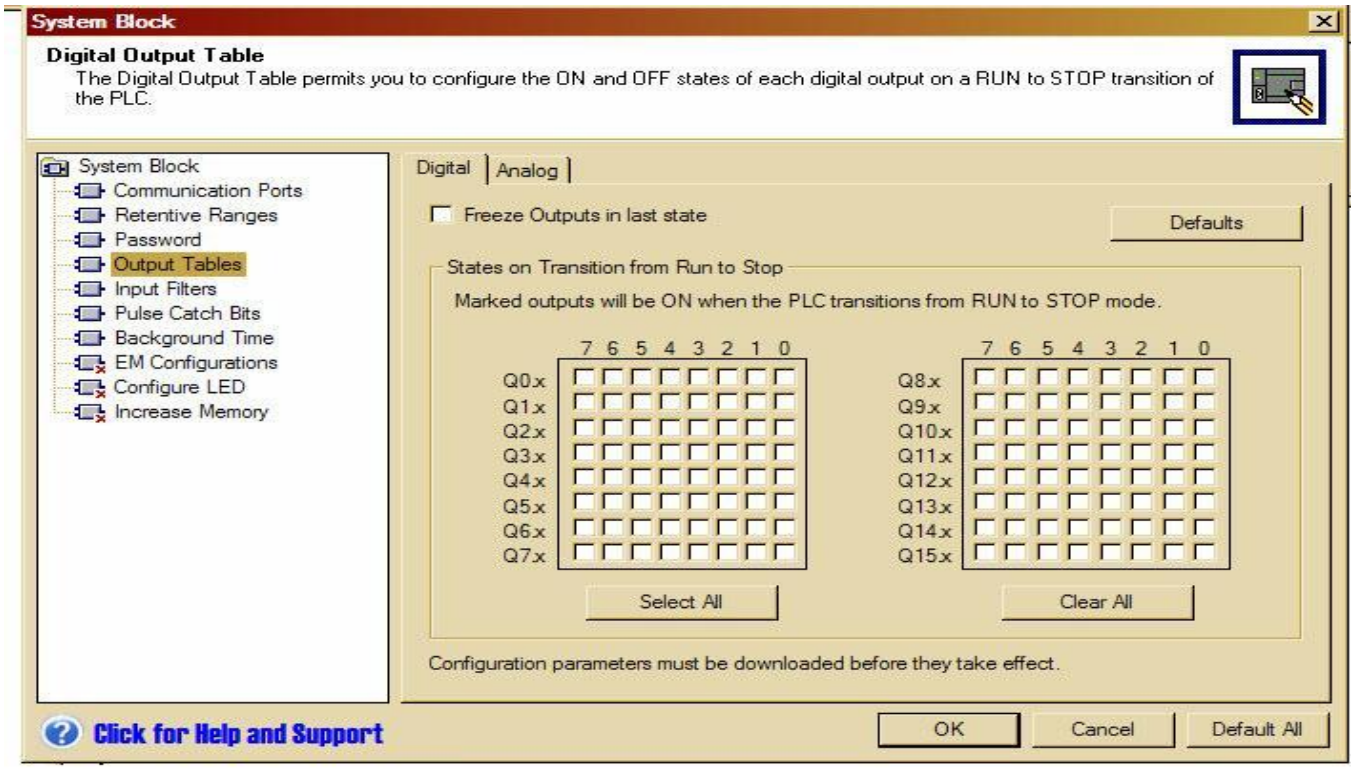
· **Level 1 (nível 1) –** Acesso total à CPU. Não será solicitada senha.

· **Level 2 –** Acesso parcial, visualização do programa e upload. A senha será solicitada para efetuar download, forçar memórias e programar.

· **Level 3 –** Acesso mínimo, visualização do programa. A senha será solicitada para efetuar upload e download, forçar memórias e programar.

· erísticas originais da CPU.

Fig. 21 – Opção Output Tables



D) Output Table (tabela de saída)

Nesta pasta obtemos recursos que nos permitem selecionar algumas saídas que serão energizadas, assim que a CPU for para o estado STOP.

Se você quiser congelar as saídas no seu último estado, escolha Freeze Outputs (congelar saídas) e clique OK.

Se você quiser copiar a tabela de valores para as saídas, entre na tabela de saídas e clique no respectivo “box” para cada saída que você quer setar “on” (1). Depois da transição da CPU de “Run” para “Stop” a mudança será confirmada. Para salvar as alterações clique OK.

Os valores default na tabela são todos zero.

OBS: Sendo a função Freeze Outputs selecionada, quando a CPU for para o estado STOP, será mantido o último estado de todas as saídas.

E) Input Filters (filtros de entrada)

Nesta pasta selecionamos um tempo que servirá de filtro, para não interpretar ruídos erroneamente nas entradas.

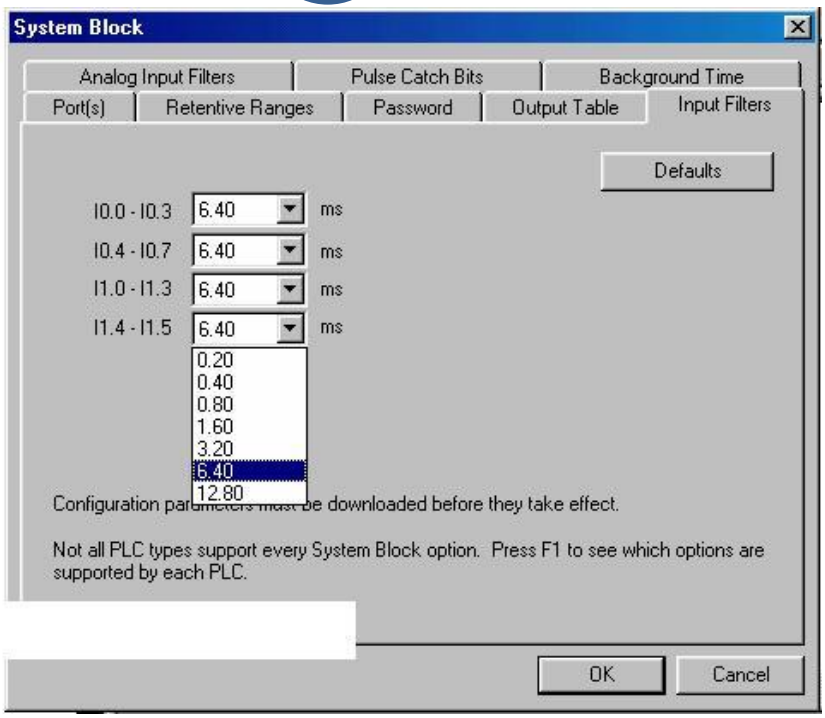


Fig. 22 – Opção Input Filters

E.1) Analog Input Filters (filtros de entrada analógica)

Nesta pasta habilitamos as entradas analógicas que estamos utilizando no projeto. Definimos o número de amostragens que devem ser feitas para executar a média e passar para o processo.

F) Background Time (tempo de retaguarda)

Nesta pasta podemos selecionar qual porcentagem do tempo de ciclo (scan) será reservada para a comunicação com placas especiais, rede, etc. O percentual “default” dedicado ao processamento de comunicação é 10%. Este valor pode ser alterado até o máximo de 50%.

sta reserva de tempo implica em termos um controle mais lento do processo.

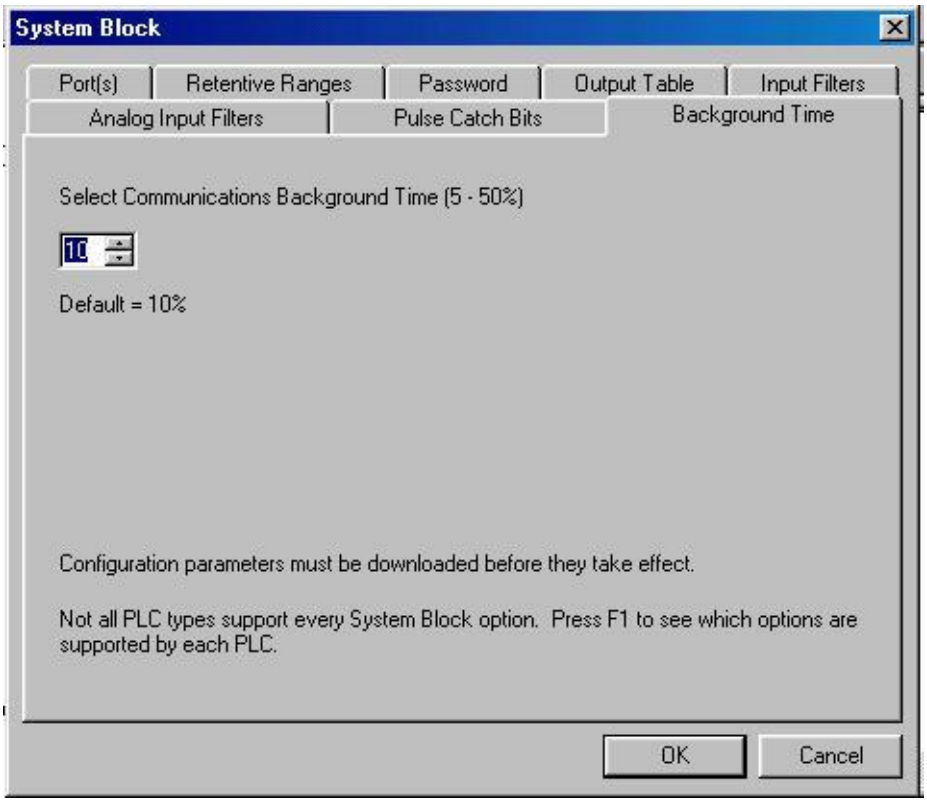


Fig. 23 – Background Time

G) PULSE CATCH BITs (BITs de captura de pulso)

Através desta pasta configuramos as entradas que deverão ser memorizadas até que a CPU inicie um novo ciclo (scan). Este recurso é muito utilizado quando uma entrada tem um tempo de estado ativo (nível lógico 1), menor que o tempo de ciclo (scan) do programa. A operação do Pulse Catch pode ser habilitada individualmente para cada entrada digital.

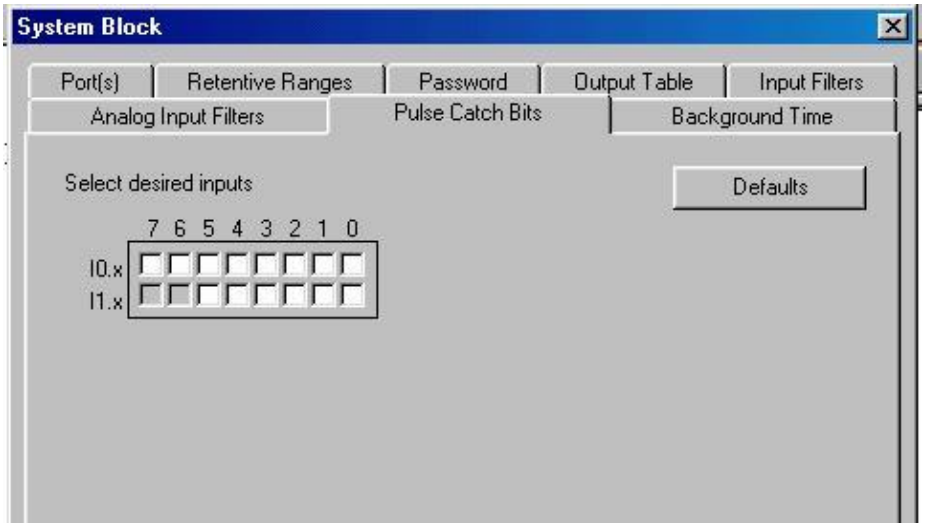


Fig. 24 – Pulse Catch Bits

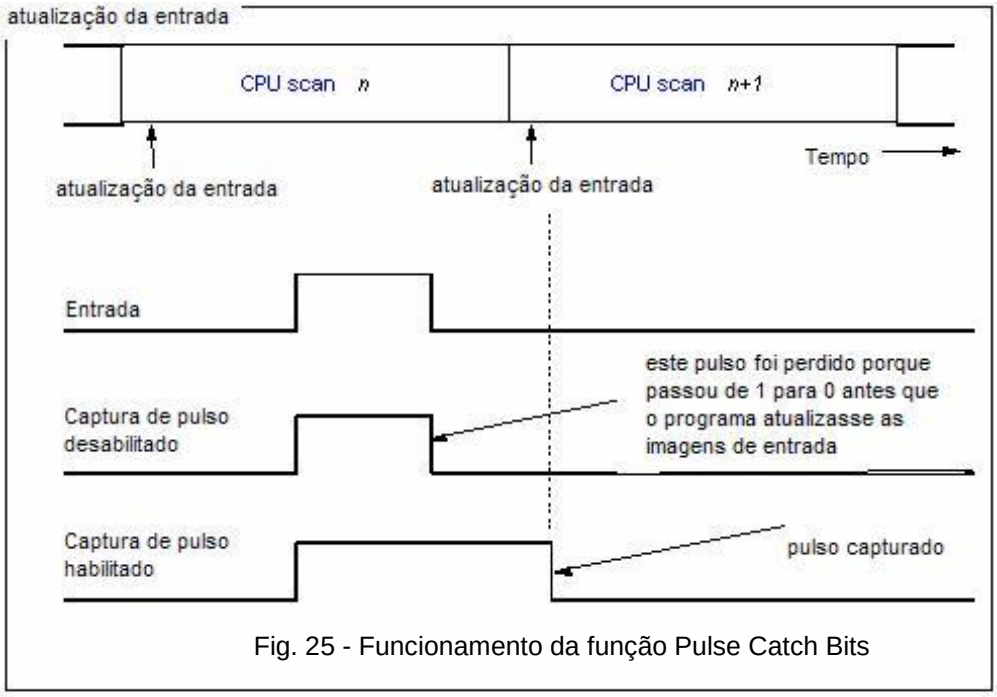

atualização da entrada
CPU scan n
CPU scan n+1
Tempo
atualização da entrada
atualização da entrada
Entrada
este pulso foi perdido porque passou de 1 para 0 antes que o programa atualizasse as imagens de entrada
Captura de pulso desabilitado
Captura de pulso habilitado
pulso capturado

Fig. 25 - Funcionamento da função Pulse Catch Bits

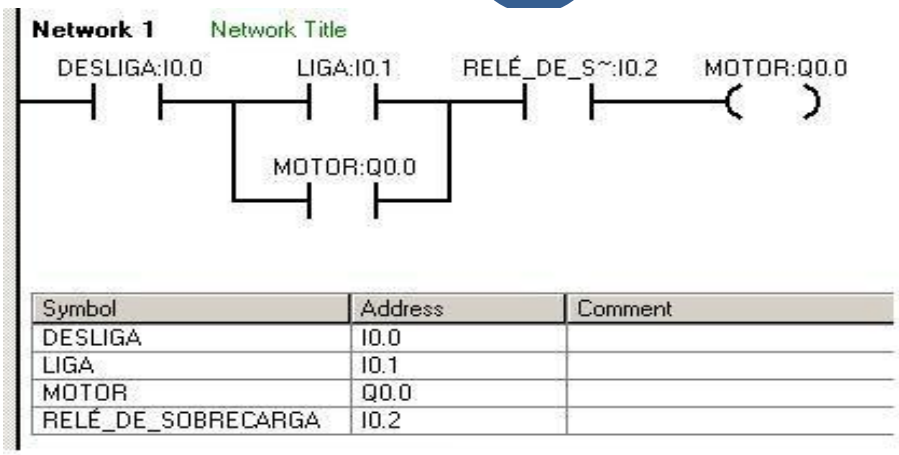
II. Program Block (bloco de programa)

No Program Block estão localizados os blocos onde o usuário realizará a programação do CLP, de acordo com as solicitações do projeto de automação.

III. Symbol Table (tabela de símbolos)

No Symbol Table podemos substituir os endereços do CLP (entradas, saídas, flags) por símbolos (texto).

Por exemplo, podemos substituir, em qualquer programa desenvolvido, a entrada I0.0 pelo símbolo DESLIGA, a entrada I0.1 pelo símbolo LIGA e assim por diante.



Symbol	Address	Comment
DESLIGA	I0.0	
LIGA	I0.1	
MOTOR	Q0.0	
RELÉ_DE_SOBRECARGA	I0.2	

Fig. 26 – Utilização da tabela de símbolos

IV. Status Chart (estado das variáveis)

No Status Chart o usuário pode verificar o status das variáveis selecionadas por ele (habilitada, desabilitada, valor da contagem, etc), bem como pode forçar o valor das referidas variáveis.

Os dados são visualizados em forma de tabela, como se pode observar a seguir.

“Tools – Options – Status Chart” – permite configurar a tela do Status Chart.

	Address	Format	Current Value	New Value
1	LIGA:I0.0	Bit	2#0	
2	DESLIGA:I0.1	Bit	2#0	
3	RELÉ_DE_SOBRECARGA:I0.2	Bit	2#0	
4	MOTOR:Q0.0	Bit	2#1	
5		Signed		

Fig. 27 – Tabela do Status Chart

Address: endereço da variável a ser observada.

Format: formato escolhido para visualizar a variável. Os formatos disponíveis são: Bit, Signed (Inteiro com sinal), Unsigned (inteiro sem sinal), Hexadecimal e Binary.

Current Value: valor atual da variável.
New Value: valor utilizado para forçar a variável.

O S7-200 permite forçar qualquer um ou todos os pontos de I/O, além disto você também pode forçar até 16 memórias internas (V ou M) ou valores de I/O analógicos (AI ou AQ). Memórias V ou M podem ser forçadas usando bytes, words ou double words. Valores analógicos só podem ser forçados usando words.

V. Cross Reference (referência cruzada)

No ícone Cross Reference é gerada uma tabela que identifica todos os operandos usados no programa.

Na tabela sãoindicados o operando (entrada, saída, memória, contador, etc), o bloco ao qual o operando pertence, a(s) network(s) na(s) qual(is) o operando está presente e a forma como o operando está sendo utilizado (contato, bobina, etc).

	Element	Block	Location	Context
1	LIGA:I0.0	MAIN (OB1)	Network 1	-|/|-
2	DESLIGA:I0.1	MAIN (OB1)	Network 1	-| |-
3	RELÉ_DE_SOBRECARGA:I0.2	MAIN (OB1)	Network 1	-|/|-
4	MOTOR:Q0.0	MAIN (OB1)	Network 1	-( )
5	MOTOR:Q0.0	MAIN (OB1)	Network 1	-| |-

Fig. 28 - Tabela do Cross Reference

VI. Communications (comunicações)

No ícone Communications testamos a comunicação entre o CLP e o computador.

Dando um clique duplo com o botão esquerdo do mouse no ícone “Double – Click to Refresh” o PC tenta estabelecer comunicação com o CLP. Quando o CLP é encontrado, a caixa de diálogo informa o endereço do mesmo na rede.

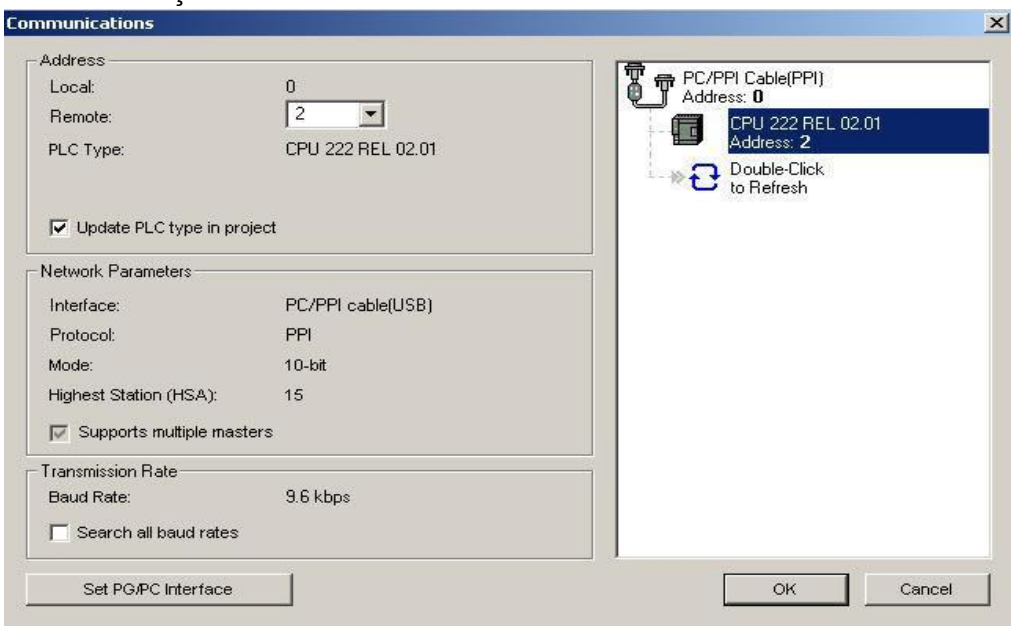


Fig. 29 – Tela do Communications

Set PG / PC interface

Neste ícone configuramos o meio físico de comunicação entre o PC e o CLP.

Neste texto será considerado o uso do cabo PPI, como meio físico de comunicação entre o PC e o CLP.

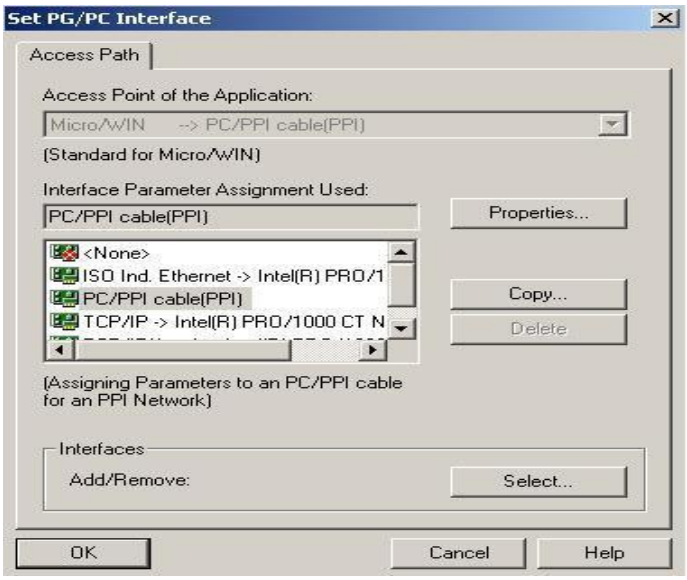
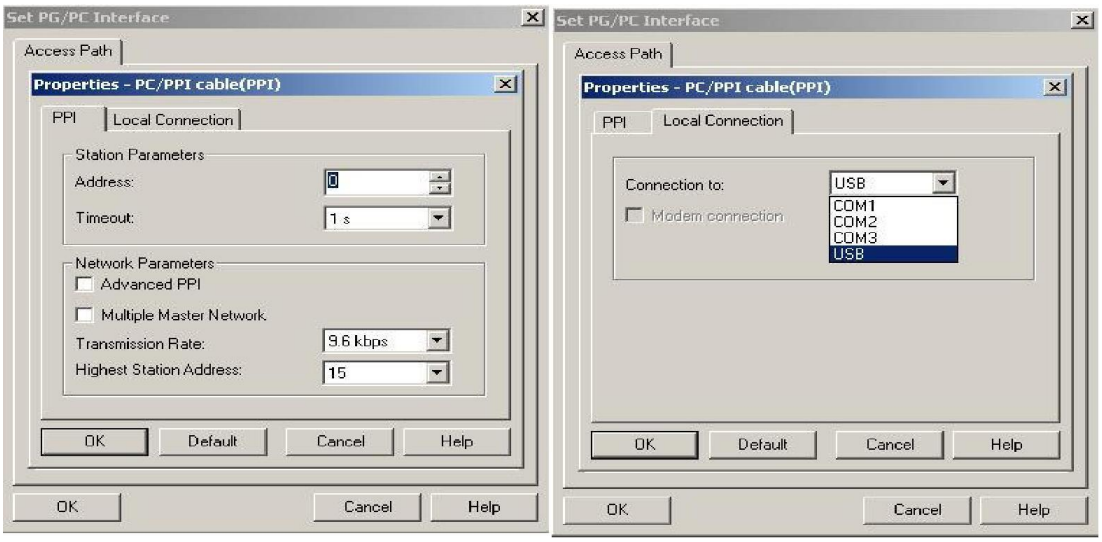


Fig. 30 – Tela Set PG/PC Interface

Na opção “Properties” configuramos o cabo PPI e o local de comunicação (portas de comunicação COM1, COM2, COM3 ou USB, neste último caso, apenas na versão V4.0 SP5 do STEP7 – Micro / Win).

Fig. 31 – Telas do Properties – PC/PPI cable



Data Block

O Data Block

é um editor de texto com forma livre.

VII. ESTRUTURA DO PROGRAMA STEP7 – MICROWIN

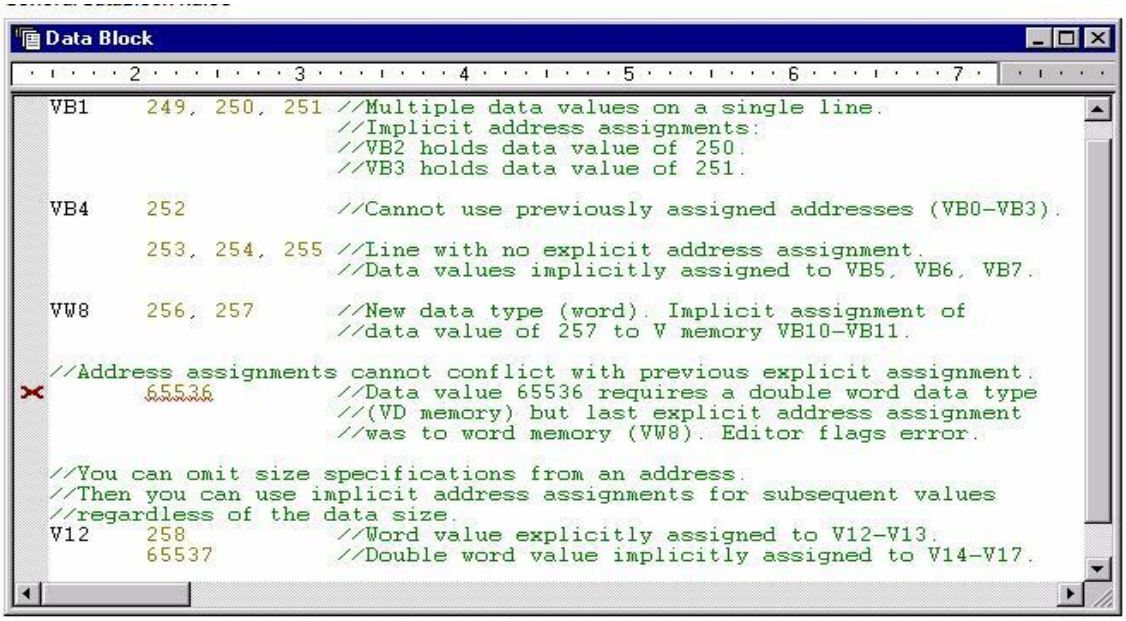

Data Block
VB1 249, 250, 251 //Multiple data values on a single line.
//Implicit address assignments:
//VB2 holds data value of 250.
//VB3 holds data value of 251.
VB4 252 //Cannot use previously assigned addresses (VB0–VB3).
253, 254, 255 //Line with no explicit address assignment.
//Data values implicitly assigned to VB5, VB6, VB7.
VW8 256, 257 //New data type (word). Implicit assignment of
//data value of 257 to V memory VB10–VB11.
//Address assignments cannot conflict with previous explicit assignment.
65536 //Data value 65536 requires a double word data type
//(VD memory) but last explicit address assignment
//was to word memory (VW8). Editor flags error.
//You can omit size specifications from an address.
//Then you can use implicit address assignments for subsequent values
//regardless of the data size.
V12 258 //Word value explicitly assigned to V12–V13.
65537 //Double word value implicitly assigned to V14–V17.

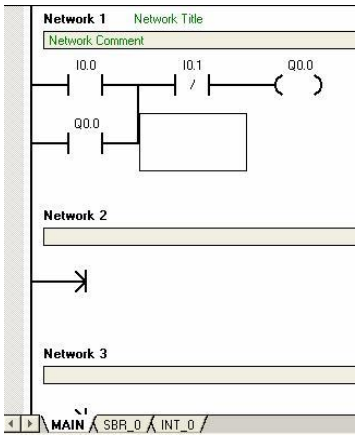
Fig. 32 – Tela do Data Block

· **Unidades Organizacionais de Programa (POU)**

OB1 (MAIN): Programa Principal. Desenvolvido pelo usuário, roda uma vez em cada ciclo (scan);

SBR_X: Sub-rotinas. Blocos desenvolvidos pelo usuário para serem executados quando habilitados por um evento programado no OB1;

INT_X: Interrupções. São blocos que podem ser desenvolvidos para serem executados a partir de um evento de interrupção.



· **Características Estruturais do Programa**

Programa Linear

Todas as instruções estão contidas em um bloco, normalmente no OB1(MAIN). Por ter todas as instruções dentro de um único bloco, deve ser usado quando temos um só programador. Todas as instruções são realizadas a cada ciclo, mesmo aquelas que não estão sendo usadas, com isto a perda de performance da CPU. Para realizar manutenção ou modificação, o programa terá de ser analisado, mesmo que a alteração seja

simples.

Exemplo:

Observar que na parte inferior esquerda da tela está ativo o MAIN (tela principal).

Programa Particionado

As instruções para cada dispositivo ou tarefa estão contidas em blocos individuais como FC ou FB. O OB1 apenas chama cada bloco em uma seqüência determinada.

No OB1 temos o programa principal e os blocos atuam como sub-rotinas do programa principal.

O programa principal e os blocos não trocam dados, porém cada área funcional tem seu bloco específico, facilitando a manutenção do programa e agilizando o processamento.

Podemos ter vários programadores, cada um programando um bloco.

OB1(MAIN) chamando bloco de sub-rotina

Exemplo:

Observe a memória SM0.0 chamando a sub-rotina 0.

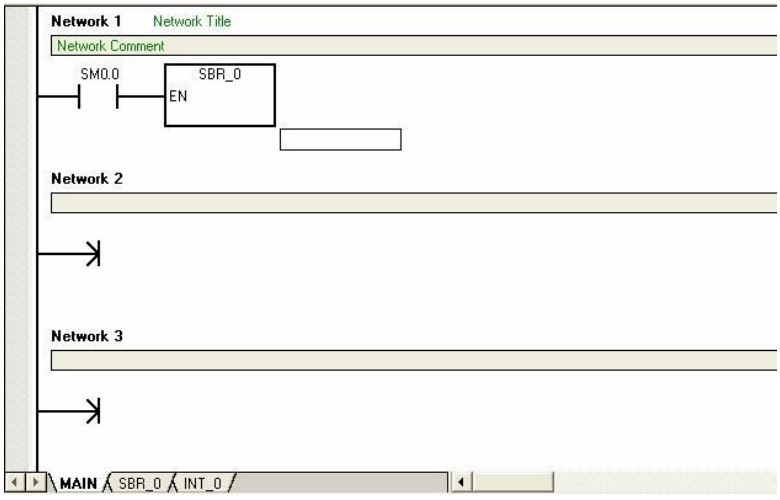


Fig. 34 – Exemplo de Programa Particionado – tela principal

E aqui, vemos a sub-rotina (SBR_0) que foi chamada anteriormente Observe o canto inferior esquerdo da tela.

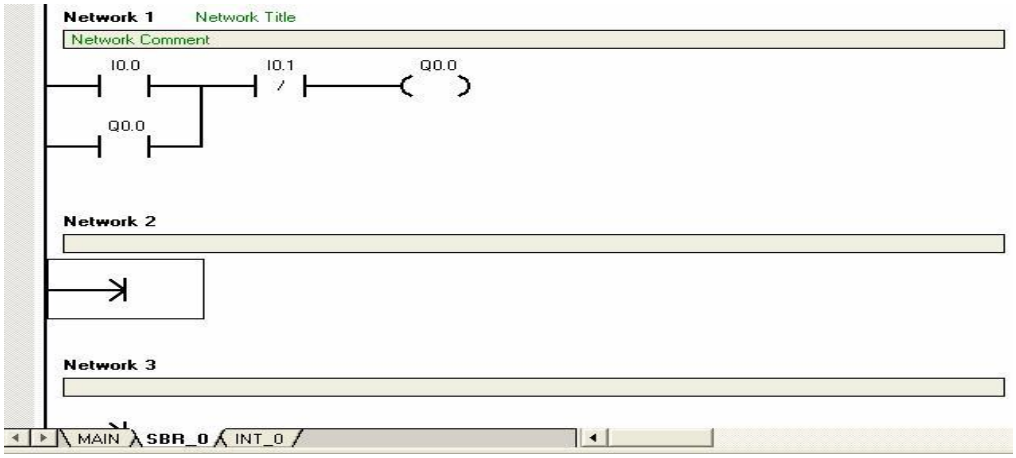


Fig. 35 – Exemplo de Programa Particionado – tela da sub-rotina

Programa Estruturado

Neste tipo de programa identificamos tipos similares ou repetitivos de funções, e criamos soluções genéricas para essas situações.

Se temos vários motores com a mesma lógica de comando, podemos criar uma lógica de comando genérica e apenas substituir os endereços específicos de cada motor.

Neste tipo de programa dados podem ser trocados.

Um exemplo do que foi dito acima está no item:
"Blocos para desenvolvimento de sub-rotinas".

· **Linguagens de Programação**

Um programa é uma série de instruções ou comandos que o usuário desenvolve para fazer com que o CLP execute determinadas ações. Uma linguagem de programação estabelece regras para combinar as instruções de forma que gerem as ações desejadas.

Há várias linguagens de programação, entretanto, a mais conhecida e tradicionalmente utilizada é a LADDER, pois se trata de uma adaptação do diagrama elétrico funcional, também conhecido como DIAGRAMA LADDER (diagramas de contatos). Como a linguagem de programação ladder é um sistema gráfico de símbolos e termos, mesmo aqueles que não estão totalmente familiarizados com os diagramas elétricos funcionais, podem aprendê-los facilmente.

Do Ladder podemos migrar para os outros modos de visualização. Nem sempre podemos fazer o contrário.

Outras estruturas de programação não tão tradicionais quanto a ladder são:

FBD = blocos lógicos (function block diagram);

STL = lista de instruções (statement list);

O STL é muito parecido com a linguagem de programação “Assembly”. Apropriado para programadores experientes.

SCL = linguagem estruturada (structured control language);

Graphset = fluxograma de um processo. Permite uma fácil compreensão do processo.

Das estruturas mencionadas, o S7-200 permite a programação em três: STL, Ladder, FDB.

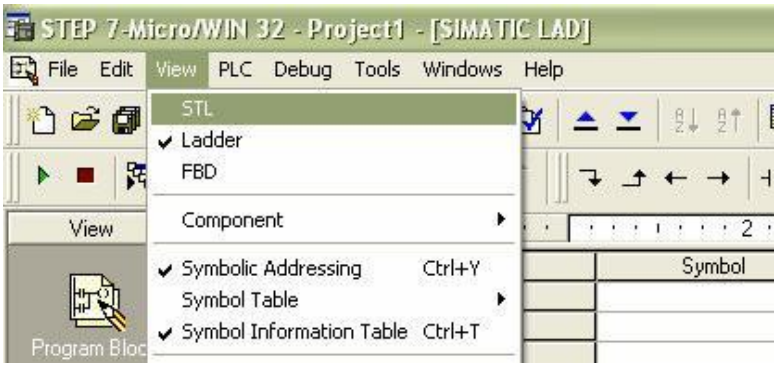
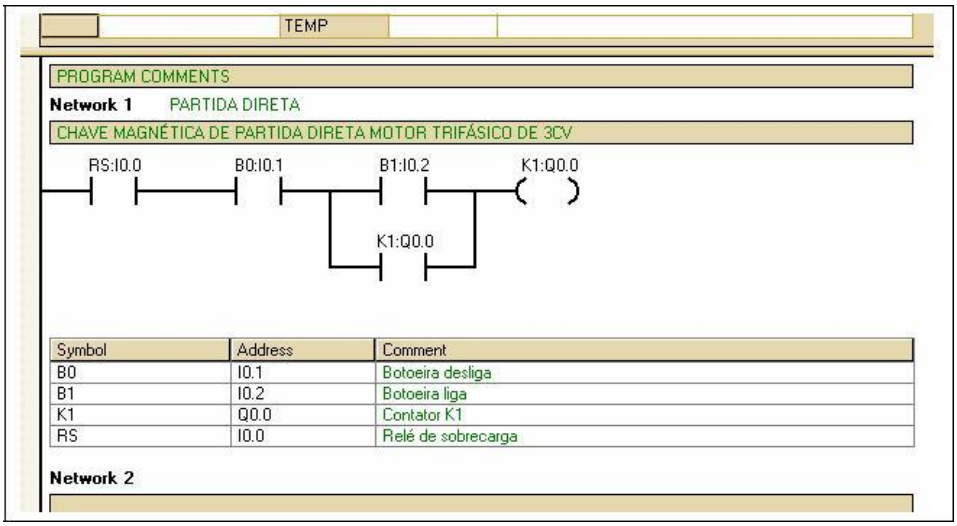


Fig. 36 – Menu View

Exemplo:

Partida direta em: Ladder

Fig. 37 – Partida Direta em Ladder



FDB

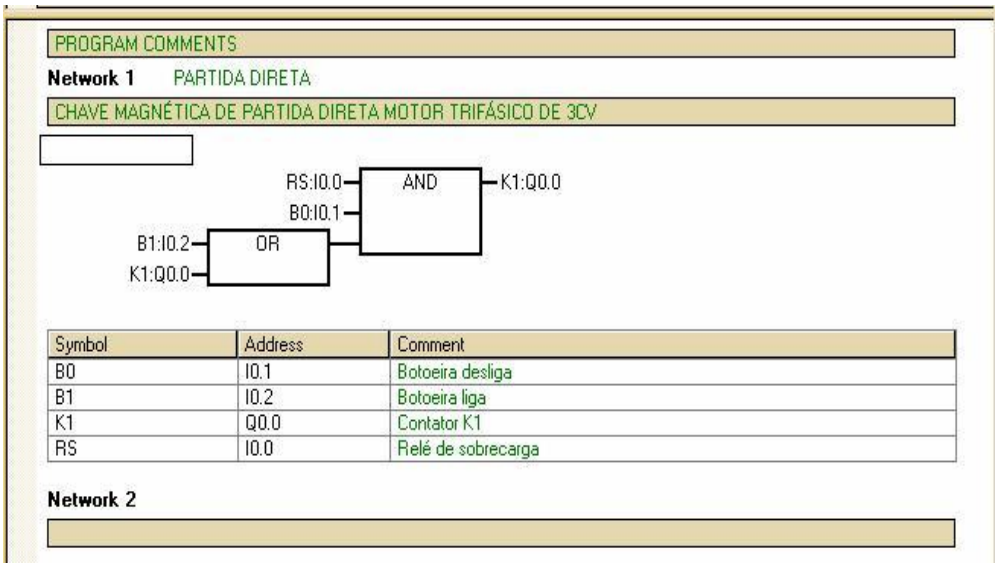


Fig. 38 - Partida Direta em FDB

STL

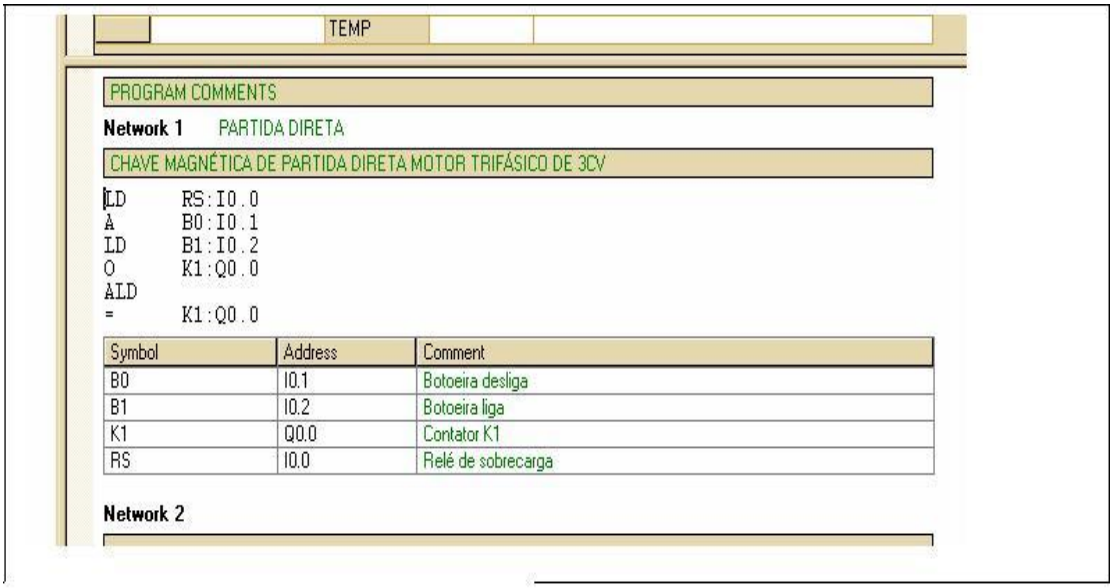


Fig. 39 - Partida Direta em STL

Network

A lógica é normalmente separada em pequenos pedaços chamados Networks. O

programa é executado uma Network por vez, da esquerda para a direita e de cima para baixo. Quando a CPU chega ao fim do programa, volta ao começo.

Cada Network só pode ter uma saída ou saídas em paralelo.
Uma saída só pode aparecer em uma Network.

Se nomear uma saída Q0.0 ela NÃO pode aparecer novamente em outra Network como saída, podendo ser usada como endereço de entrada, fazendo que a ação desta Network esteja condicionada à ação anterior.

Este endereço pode ser usado em uma entrada para fazer o “pega” de um motor, por exemplo, e neste caso podemos usar o mesmo endereço em várias Networks.

Tipos de Memória

Uma memória é uma entidade virtual que é utilizada apenas para ajudar o desenvolvimento da lógica de programação escalar interna. Ela usa a mesma simbologia utilizada para entrada e saída.

O S7-200 armazena informações em diferentes localizações de memória. Você pode acessar dados na CPU em vários tipos de área de memória (V, I, Q, M, S, L, e SM) como bytes, words, ou double words. Para acessar um dado no formato de byte, word, ou double word você deve especificar o endereço.

IMPORTANTE!!!

Endereço iniciado com M (memory) é virtual e substitui, por exemplo, os contatos auxiliares. A memória do tipo M tem um range pequeno (do byte 0 ao byte 31).
Endereço iniciado com V também é virtual, como VM. A memória tipo V tem range bem maior (byte 0 ao byte 2047); sendo assim é interessante usar a memória V.

Endereçando uma variável na memória - V

Você pode usar a memória V para armazenar resultados intermediários de operações que são executadas pela lógica de controle em seu programa, ou para armazenar outros dados que pertencem a seu processo ou tarefa. Você pode ter acesso à memória V em bits, bytes, word ou double words.

bit address = V10.2

word address = VW100 (usando os bytes 100 e 101)

Endereçando uma variável na memória – M

Você pode usar a memória M para armazenar o estado intermediário de uma operação ou outra informação de controle. Você pode ter acesso à memória M em bits, bytes, word ou double words.

Bit address = M26.7.

Double word address = MD20 (usando os bytes de 20 a 23).

Endereçando uma memória especial – SM

Os bit SM propiciam um meio de comunicação entre a CPU e seu programa. Você pode usar esses bit para selecionar e controlar algumas das funções especiais do S7-200:

Um bit que é 1 para o primeiro ciclo do scan.

Um bit que mostra o status das instruções de operação e das instruções matemáticas.

Bit SM address = SM0.1

Byte SM address = SMB86

Memória Local e Global

É similar à memória “V” com uma exceção. A memória “V” tem um escopo global, enquanto a memória “L” tem um escopo local.

O termo escopo global significa que o mesmo local de memória pode ser acessado por qualquer entidade do programa principal, sub-rotina ou rotina de interrupção.

O termo escopo local significa que a alocação de memória está associada com a entidade de programa em particular.

Você pode acessar a memória L como bit, word ou double word.

Bit L address = L0.0.

Byte L address = LB33.

IMPORTANTE!!!

As variáveis globais estão associadas às áreas de memória que são usadas pelo CLP. As memórias podem ser I0.0, I0.1,
...,Im.n, Q0.0, Q0.1, Qm.n, V0.0, V0.1, ..., Vm.n, M0.0, M0.1,
...,Mm.n, etc. O uso dessas memórias é sempre único, pois uma vez utilizada em um rotina ou sub-rotina do programa, ela NÃO poderá ser utilizada em outra ou até na própria rotina/sub-rotina. O que é declarado na Variable Table é sempre variável global.

As variáveis locais são aquelas que são válidas apenas para a rotina que está sendo programada, e são declaradas na tabela que surge no topo da janela OB1 e demais sub-rotinas.

Quando usar variáveis globais ou locais?

A utilização de variáveis globais é mais comum, pois normalmente os programas são feitos para uma determinada aplicação ou máquina.
O uso de variáveis locais é indicado em rotinas criadas para utilização em diversas aplicações, por exemplo: uma sub-rotina para uma chave YDELTA, que poderá ser utilizada em diversas máquinas.

Variáveis temporárias - TEMP

O tipo de variável local que você pode usar depende do POU "Unidades Organizacionais de Programa", onde você está.

O programa principal OB1, as rotinas de interrupção e as sub-rotinas podem usar variáveis temporárias (TEMP).

Variáveis temporárias só estão disponíveis enquanto o bloco está sendo executado e estão prontas para serem reescritas, quando a execução do bloco estiver completa.

Sub-rotinas podem também ser usadas para chamar parâmetros (IN, IN_OUT, OUT).

IN - parâmetro de
entrada; OUT -
parâmetro de saída;

IN_OUT – parâmetro cujo valor é suprido pela POU, modificado pela sub-rotina, retornando para a POU.

TEMPORARY - variável temporária que é salva temporariamente na pilha de dados locais. Uma vez que a POU seja executada completamente, o valor da variável temporária não está mais disponível.

Variáveis temporárias não guardam seu valor entre as execuções da
POU.

Endereçando um acumulador – AC

Acumuladores são equipamentos de leitura e escrita que podem ser usados como memória.

Você pode usar acumuladores para passar parâmetros de uma sub-rotina e armazenar valores intermediários usados no cálculo.

A CPU propicia 4 acumuladores de 32 bits (AC0, AC1, AC2 e AC3). Você pode acessar os dados dos acumuladores como bytes, words ou double words.

Endereçando um contador de alta velocidade – HC

Esses contadores contam eventos em alta velocidade, de forma independente do tempo de scan da CPU.

São acessados por uma memória do tipo HC, podendo ser endereçados apenas como double word (32 bits).

VIII. PROJETANDO NO S7-200

Criando um Projeto no S7-200

Divida seu processo em seções que tenham um nível de independência uma da outra.

Escreva a descrição da operação de cada seção do processo ou máquina:

- Pontos de I/O;
- Descrição da operação;
- Estados permissivos (estados que devem ser alcançados antes de permitir ação) para cada actuator (solenóides, motores, etc.);
- Descrição da interface de operação;
- Interface com outras seções do processo ou máquina;
- Desenho dos circuitos de segurança;
- Identificar equipamentos requeridos pela lógica de segurança.

Fazendo Tools – Options – General podemos selecionar como vamos trabalhar, em termos de mnemônicos para exibição das instruções no editor de programa.

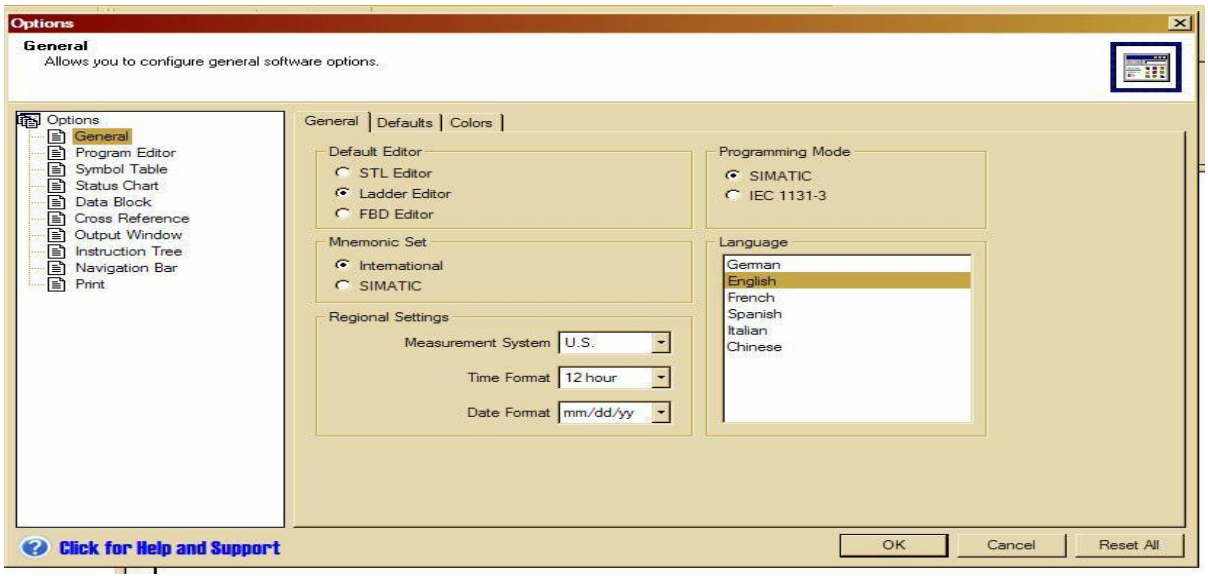


Fig. 40 - Menu – Tools – Options

Podemos escolher entre:

Simatic – Simatic, que utiliza os mnemônicos em alemão, ou seja, entrada E (Eingabe) e saída A (Ausgabe).

Simatic – Internacional, mais usual, que usa os mnemônicos em inglês, I (Input) e Q (Quit).

IMPORTANTE!!!

Saída em inglês comumente seria “Output”, mas usar o mnemônico “O” criaria confusão com o número zero, daí o uso do “Quit”.

Para criar um projeto para ser aplicado no CLP S7-200, devemos seguir os seguintes passos:

1. Abrir o programa STEP7-Micro/Win no micro;
2. Selecionar o item File na barra de tarefas;
3. Em seguida o subitem New;
4. Ou clicar sobre o item folha em branco na barra de atalhos;
5. Depois de ter criado o projeto seguindo os passos do item anterior, selecionar agora o tipo de CPU a ser utilizada.

· Selecionar o item CLP na barra de tarefas e em seguida o subitem Type.

Em seguida surgirá uma janela, onde será escolhido o tipo da CPU no item CLP Type. Se o CLP já estiver conectado no micro por meio do cabo de conexão, basta clicar no item Read CLP que o sistema reconhecerá o tipo de CPU.

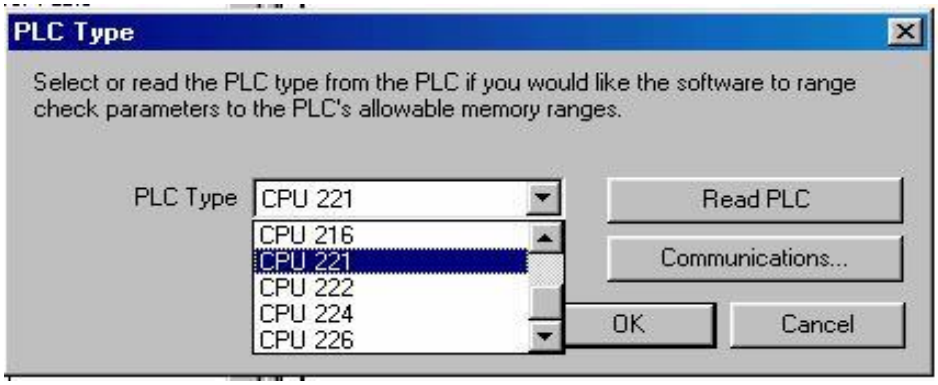


Fig. 41 – Tela PLC Type

· Após essa seqüência já podemos iniciar o projeto propriamente dito.

Observe que até esta etapa o projeto só foi criado e não desenvolvido. Em breve, estaremos desenvolvendo o projeto em linguagem de programação Ladder com os recursos oferecidos pelo S7-200.

PARA EXECUTAR O PROGRAMA NO PLC

- fazer a lógica no PC no ambiente do step – 7;
- salvar;
- compilar;
- download para o PLC;
- colocar o PLC em RUN via PC;
- ativar "PROGRAM STATUS" que permite ver funcionamento do programa;
- atuar as chaves físicas para produzir funcionamento.

Depois de ter criado um novo projeto e realizado a lógica de programação, por intermédio dos blocos disponíveis no CLP S7-200, chegou a hora de estabelecermos a comunicação do CLP com o PC para que possamos transferir o projeto desenvolvido no PC para o CLP.

1º Passo: selecionar na barra de ferramentas o item CLP e depois o subitem Type.

2º Passo: surgirá uma nova janela, onde deveremos selecionar a opção Communications.

3º Passo: após selecionar a opção Communications, surgirá a janela Communications Setup, onde devemos escolher a opção PC / PPI cable (PPI).

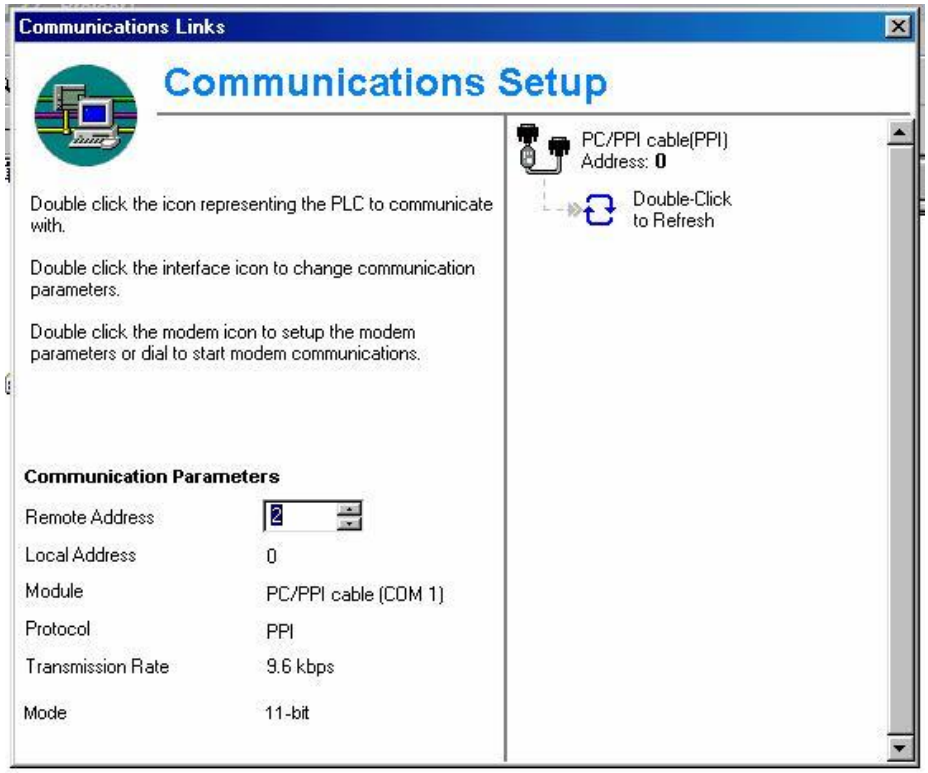


Fig. 42 – Tela Communications Setup

4º Passo: surgirá a janela Set PG / PC Interface; nesta janela escolheremos o tipo de comunicação do CLP com o PC ou rede de CLP. Para o nosso caso a comunicação será via cabo PPI, opção PC/PPI cable (PPI).

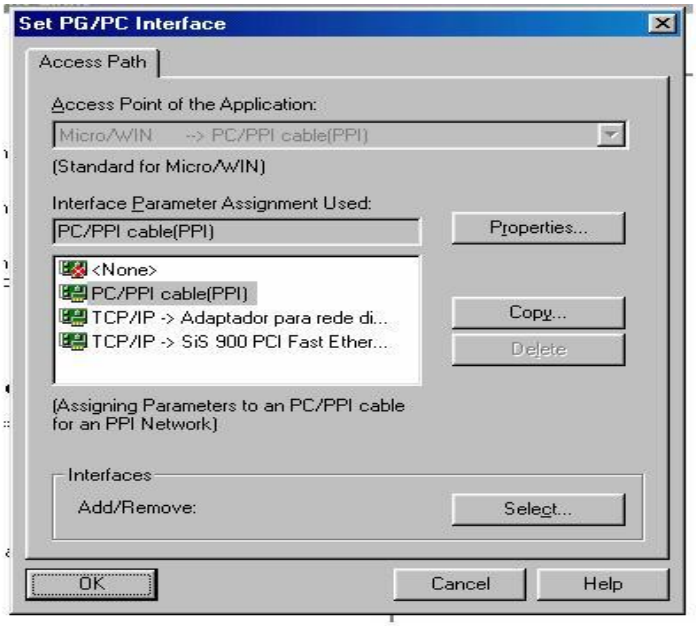
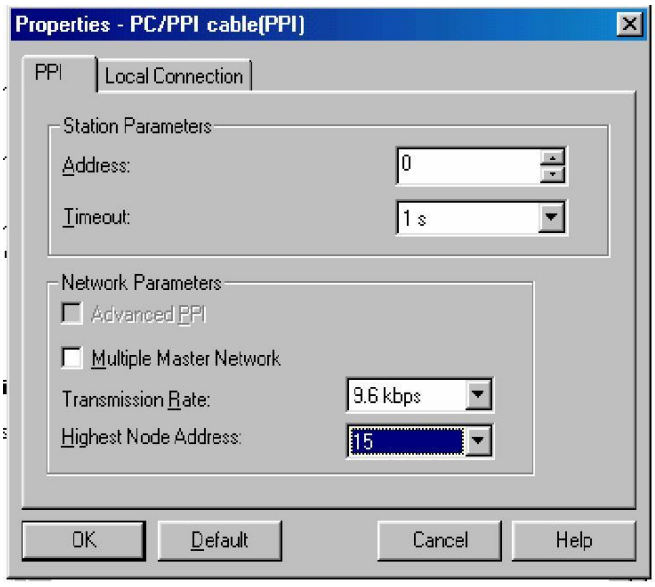


Fig. 43 – Tela Set PG/PC Interface

5º Passo: após ter escolhido o meio de comunicação do CLP com o PC, que foi a opção PC/PPI cable (PPI), deveremos agora selecionar a opção Properties..., para configurarmos a velocidade de comunicação, o endereço do CLP na rede, a porta de comunicação do computador (COM1 ou COM2), etc.



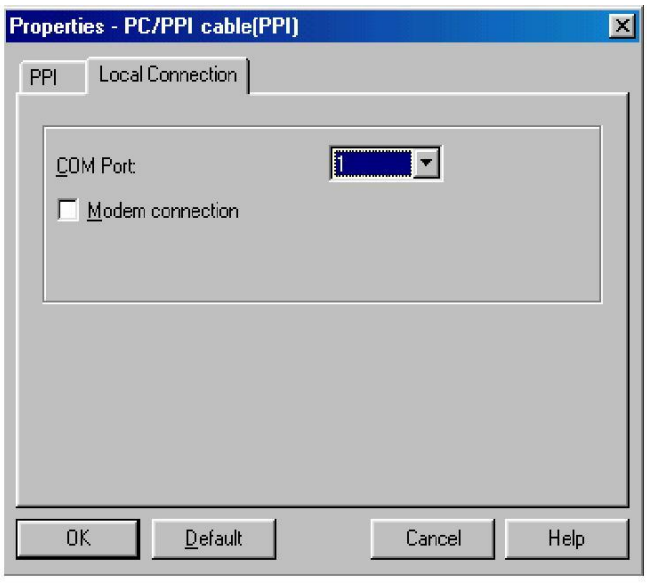


Fig. 44 – Telas Properties PC/PPI cable

6º Passo: depois de configuradas as opções da janela Properties – PC/PPI cable (PPI), devemos clicar no botão OK, passando para a próxima janela.

7º Passo: quando a janela surgir, deveremos dar um click duplo na opção Double Click to Refresh. Se a comunicação estiver correta aparecerá o modelo da CPU do CLP.

8º Passo: fechar as janelas e retornar à tela principal.

Transferindo o Projeto do PC para o CLP

Após ter estabelecido a comunicação do PC com o CLP, nossa última etapa consiste na transferência do projeto desenvolvido no PC para o CLP. No caso do S7-200, esta etapa é chama de download. Antes de se realizar o download faz-se necessário que se realize a compilação do programa, para verificação de alguma falha com relação à utilização dos blocos do S7-200. A compilação pode ser realizada utilizando-se a tecla localizada na barra de atalhos.

Realizando o Download

1º Passo: abrir o projeto que se deseja transferir para o CLP;

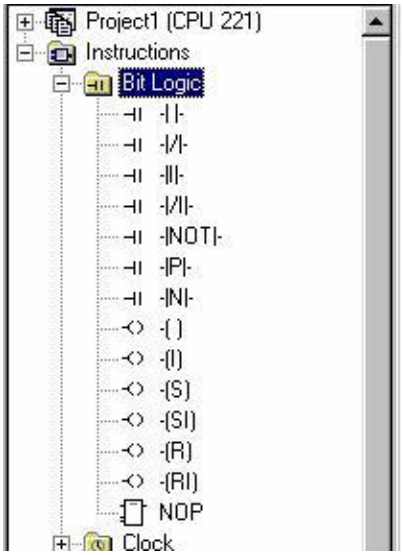
2º Passo: selecionar a opção File da barra de ferramentas e a sub- opção Download, ou a tecla na barra de atalhos;

3º Passo: Se não houver nenhuma falha na comunicação, o projeto será transferido normalmente.

Depois de ter sido realizado o download basta agora realizar as simulações para verificar a eficácia do projeto desenvolvido, caso haja algo a ser modificado na programação do CLP é na etapa de simulação que isto ficará mais claro.

IX. PASTA DE INSTRUÇÕES

Nesta pasta encontraremos todos os elementos necessários para o desenvolvimento dos projetos.



1. Instruções Binárias
2. Temporizadores
3. Contadores
4. Comparadores
5. Blocos de movimentação de dados
6. Operações matemáticas
7. Conversores
8. Blocos para desenvolvimento de sub-rotinas

Fig. 45- Menu Bit Logic

1. Instruções Binárias

Sinal Digital

As grandezas físicas, às quais são atribuídos unicamente dois valores ou níveis, são chamadas de grandezas digitais ou sinais binários.
Exemplo de sinal digital: contato aberto ou fechado de uma botoeira ou relé de sobrecarga.

Estas instruções estão contidas na pasta Bit Logic.
São instruções relacionadas a bits, ou seja, dois únicos estados: 0 ou 1.
Nesta pasta encontramos os contatos, as bobinas, as instruções de set e reset, os pulsos P (borda positiva ou de subida) e N (borda negativa ou de descida) e a instrução Not.

Parametrização:

No CLP S7-200 as entradas são designadas pela letra I (input) seguida de dois números; o primeiro se refere ao Byte e o segundo ao Bit.
Exemplo: I 0.7 (entrada - bit 7 do byte 0)

As saídas são designadas pela letra Q (quit) também seguida de dois números.
Exemplo: Q 3.2 (saída – bit 2 do byte 3)

Contato (entrada)

O CLP S7-200 dispõe de dois tipos de contatos: contato scan em 1 e contato scan em 0.

O contato scan 1 funcionará de acordo com o sinal de seu respectivo endereço, ou seja, se o endereço do contato estiver no nível lógico 1, o contato também estará no nível lógico 1.

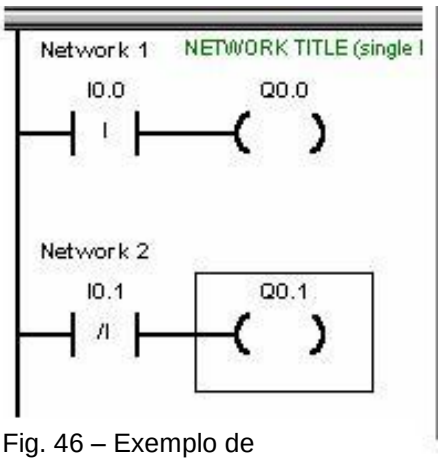


Fig. 46 – Exemplo de utilização do contato

O contato scan 0 funcionará de forma oposta a de seu endereço, ou seja, se o endereço do contato estiver no nível lógico 1, o contato estará no nível lógico 0.

O CLP S7-200 possui também dois contatos especiais que são imediatos, ou seja, não esperam o final da varredura para atualizar seu status. Estes contatos são utilizados para instruções de emergência, quando não se pode esperar todo o tempo de execução do ciclo de varredura.

Bobina (saída)

A bobina é energizada quando o resultado lógico formado pelos contatos e outras instruções antecedentes à mesma, for igual a 1.

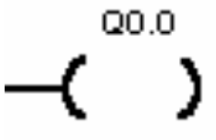


Fig. 47 - Bobina

Da mesma forma que os contatos, também existem bobinas especiais que são de atuação imediata, ou seja, não esperam o final da varredura para atualizar.

Instrução de Set e Reset

Nestes tipos de bobinas não há a necessidade que a lógica antecedente a elas seja sempre igual a 1, basta uma varredura para que a bobina energize (Set) ou desenergize (Reset).

O número na parte inferior da bobina indica quantos bits, a partir do endereço inicial, o programa irá “setar” ou “resetar”.

Também existem bobinas do tipo set ou reset Imediato.

Vamos testar? Execute a rotina, a seguir, no CLP e analise o resultado

Partida direta utilizando as bobinas de set e reset.

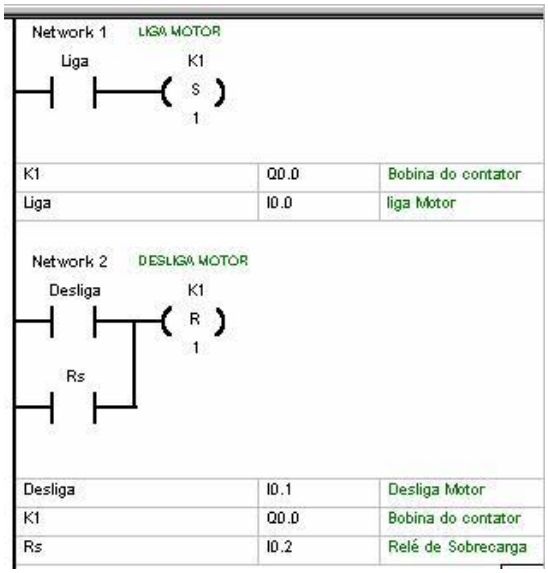


Fig. 48 – Exemplo de utilização de set-reset

Pulsos – P / N

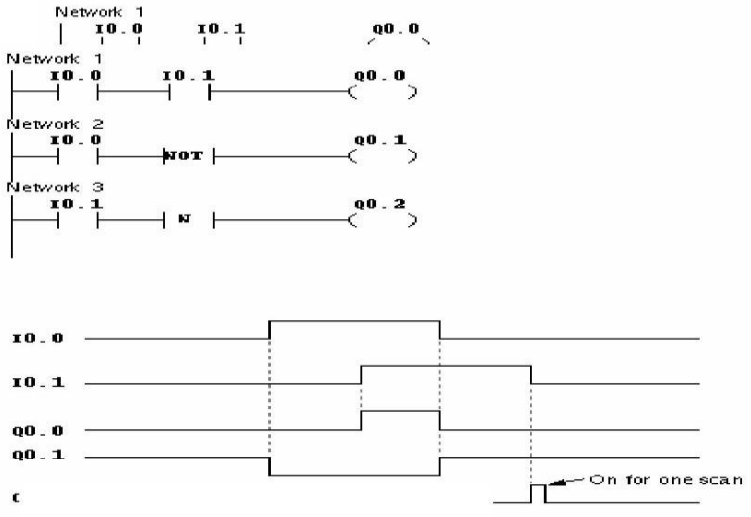
São contatos que detectam bordas de subida (P) ou descida (N) da lógica anterior a eles, ficando no nível lógico 1 por uma varredura (scan), logo em seguida retornam ao nível lógico 0.

Instrução NOT

Esta instrução inverte o resultado lógico da lógica de programação anterior a ela, ou seja, se o resultado lógico da lógica de programação anterior a ela for 0, ela transforma em 1, e vice versa.

Vamos testar? Execute a rotina, a seguir, no CLP e analise o resultado

Fig. 49 – Tela Set PG/PC Interface



Vamos praticar? Objetivo: transformar a lógica tradicional de relés em Ladder.

Exercício – Partida Direta

Elaborar no CLP uma rotina de programação linear, que atenda as condições do circuito auxiliar de comando por partida direta, para um motor de indução trifásico. Faça o programa em Ladder e depois converta para STL e FDB.

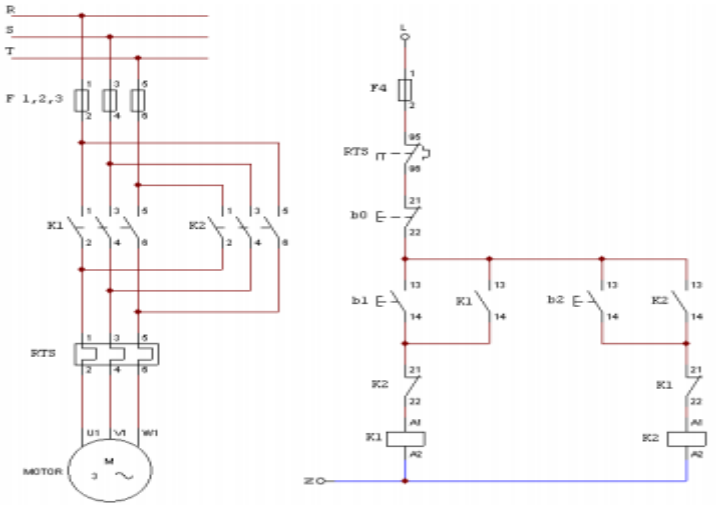


Fig. 50 – Diagrama de força e comando da partida direta com reversão

Vamos praticar? Objetivo: transformar a lógica tradicional de relés em Ladder.

Exercício – Partida Direta com Reversão

Elaborar no CLP uma rotina de programação particionada, que atenda as condições do circuito auxiliar de comando para uma partida direta com reversão de um motor de indução trifásico. Faça o programa em ladder e depois converta para STL e FDB.

Vamos praticar? Objetivo: exercitar a utilização do sensor digital para informação de posição.

Exercício – Prensa para dobrar chapas

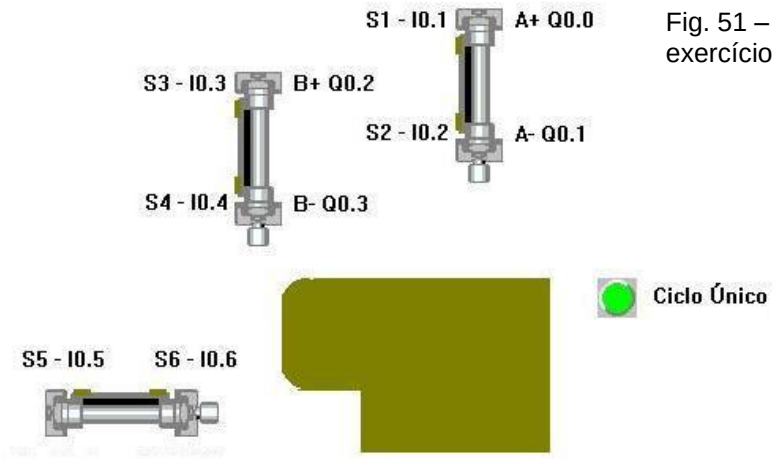


Fig. 51 – Diagrama de simulação do exercício prensa para dobrar chapas

Elabore a rotina de programação em Ladder, de forma particionada, de acordo com as orientações a seguir:

· O processo de dobramento de chapas será iniciado pela botoeira liga. Quando a mesma for pressionada, o cilindro “A” deverá avançar para fixar a chapa na mesa de dobramento;

· Quando a chapa estiver fixada na mesa de dobramento (cilindro A avançado) o cilindro “B” deverá avançar para realizar a primeira dobra na chapa;

· Após o cilindro “B” ter avançado e realizado a primeira dobra na chapa, o mesmo deverá permanecer avançado e acionar o avanço do cilindro “C” para que este possa realizar a segunda dobra na chapa;

 Quando a segunda dobra da chapa tiver sido realizada (cilindro C avançado), os três cilindros devem voltar ao estado inicial para que o processo de dobramento de chapas possa ser retomado.

Faça um diagrama elétrico indicando as conexões, no CLP, dos dispositivos de entrada e saída do processo.

Temporizadores

Estas instruções estão contidas na pasta “Timers”.

O Step 7 – 200 dispõe de três tipos de temporizadores:

· TON – Temporizador ao trabalho, ou com retardo na energização;

· TOF – Temporizador ao repouso, ou com retardo na desenergização;

· TONR – Temporizador ao trabalho com retenção, ou com retardo na energização com retenção.

Esses temporizadores possuem endereços específicos, para cada tipo e resolução de contagem, de acordo com a tabela a seguir.

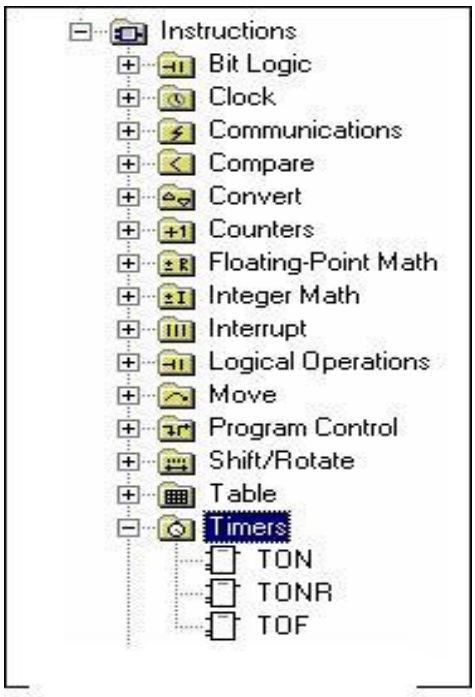


Fig.52 – Menu Instructions

Tipo do Temporizador	Resolução	Valor Máximo	Número do Temporizador
TONR	1 ms	32.767 s	TO, T64
	10 ms	327.67 s	T1-T4, T65-T68
	100 ms	3276.7 s	T5-T31, T69-T95
TON, TOF	1 ms	32.767 s	T32, T96
	10 ms	327.67 s	T33-T36, T97-T100
	100 ms	3276.7 s	T37-T63, T101-T255

Tabela 04 – Tipos de temporizadores

Para especificar o tempo de atuação do temporizador deve-se escolher um valor de resolução, que pode ser 1, 10 ou 100 ms, dependendo do endereço do temporizador escolhido, e a constante de contagem (PT) que deve ser um número inteiro.

Por exemplo: se desejamos utilizar um temporizador para a contagem de 8 segundos poderemos escolher a resolução de 100 ms e a constante de contagem 80, ou a resolução de 10 ms e a constante de contagem de 800.

Temporizador – TON

Quando a entrada IN estiver no nível lógico 1 a contagem de tempo será iniciada. Após atingido o valor de contagem estabelecido em PT, o endereço do temporizador irá para o nível lógico 1.

Deve-se escolher uma entrada (IN) responsável pela ativação da contagem e a constante de contagem (PT).

Sempre que a entrada IN do temporizador for para o nível lógico 0, o valor de tempo contado será zerado e o endereço do temporizador irá para o nível lógico 0, caso tenha conseguido ir para o nível lógico 1.

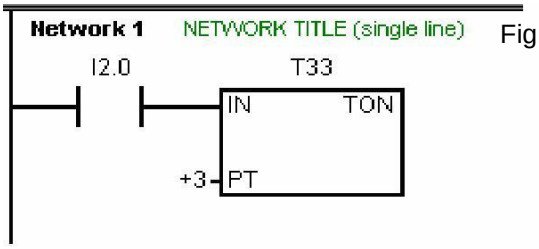
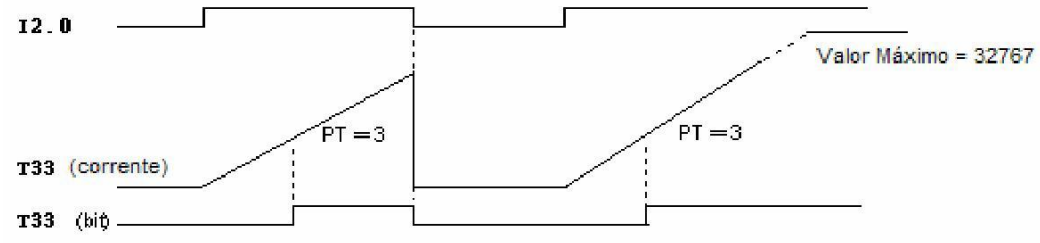


Fig. 53 – Funcionamento do temporizador TON



Vamos testar? Execute a rotina a seguir no CLP e analise o resultado

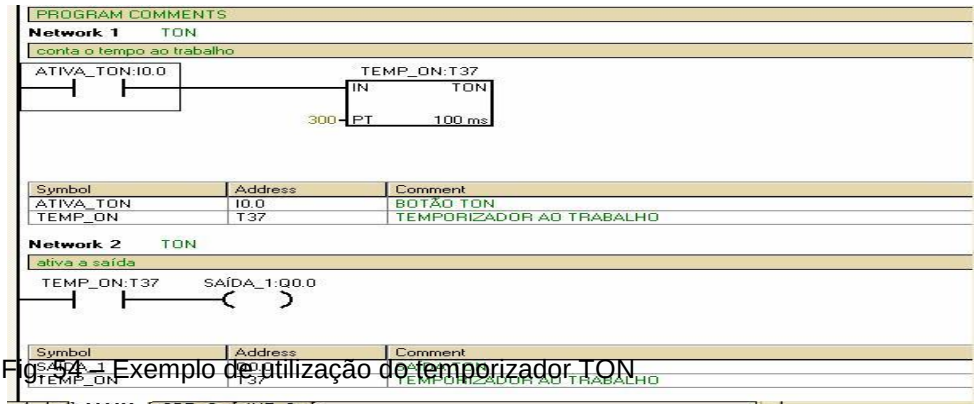


Fig. 52 – Exemplo de utilização do temporizador TON

Temporizador – TOF

Quando a entrada IN do temporizador for para o nível lógico 1 o endereço do temporizador também irá para o nível lógico 1. Quando a entrada IN do temporizador passar do nível lógico 1 para o nível lógico 0 será iniciada a contagem de tempo programado em PT e, quando este valor for atingido, o endereço do temporizador irá para nível lógico 0. Se a entrada IN voltar a 1 antes de concluída a contagem do tempo determinado, o endereço do temporizador continuará em nível lógico 1.

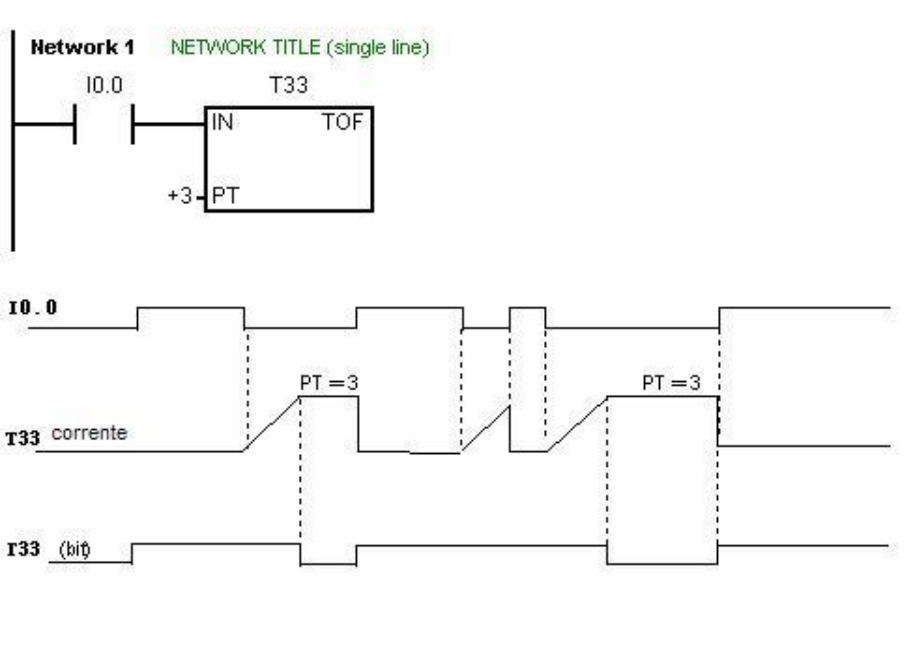


Fig. 55 - Funcionamento do temporizador TOF

Vamos testar? Execute a rotina a seguir no CLP e analise o resultado

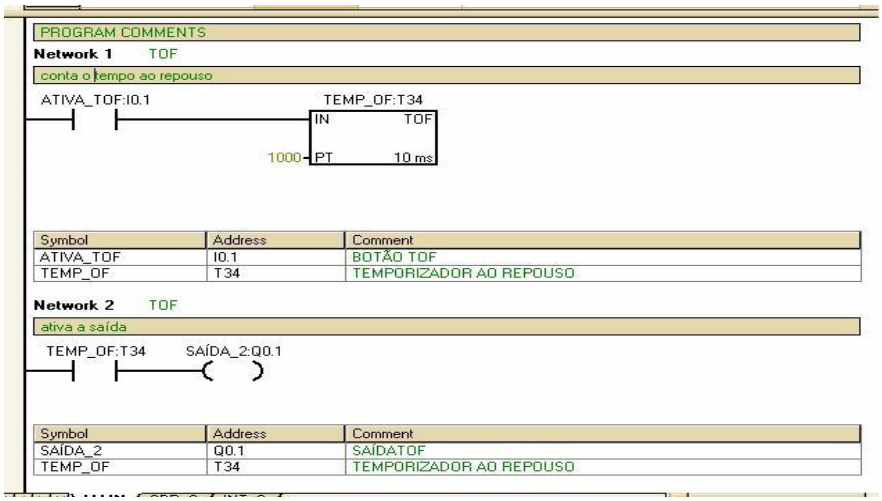


Fig. 56 - Exemplo de utilização do temporizador TOF

Temporizador – TONR

Deve-se escolher uma entrada (IN) responsável pela ativação da contagem do temporizador e a constante de contagem (PT). Quando a entrada IN estiver no nível lógico 1, a contagem de tempo será iniciada, se a referida entrada for para o nível lógico 0 o tempo já contado ficará armazenado. Quando a entrada IN for novamente para o nível lógico 1 a contagem recomeçará a partir do valor que ficou armazenado.

Quando o valor de PT for atingido, o endereço do temporizador irá para o nível lógico 1. Para que se possa mandar o endereço do temporizador para o nível lógico 0, uma vez atingido o valor de pré-set, devemos utilizar uma bobina de reset

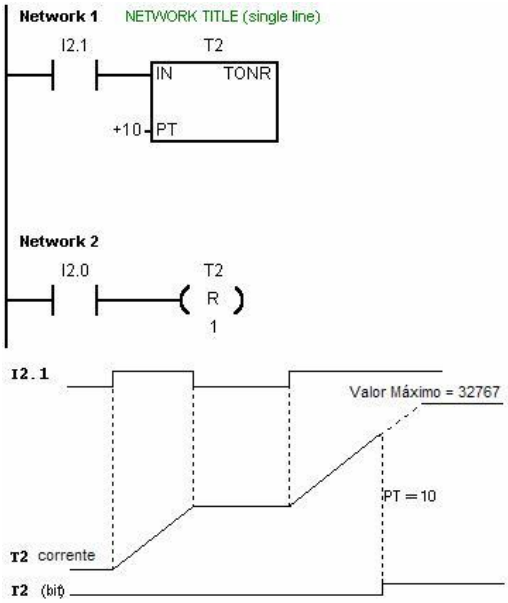


Fig. 57 - Funcionamento do temporizador TONR

Vamos testar? Execute a rotina a seguir no CLP e analise o resultado

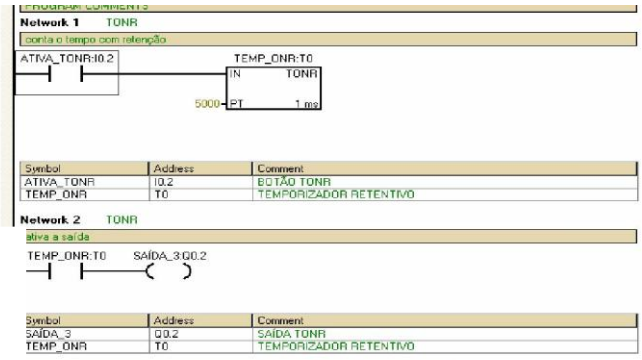
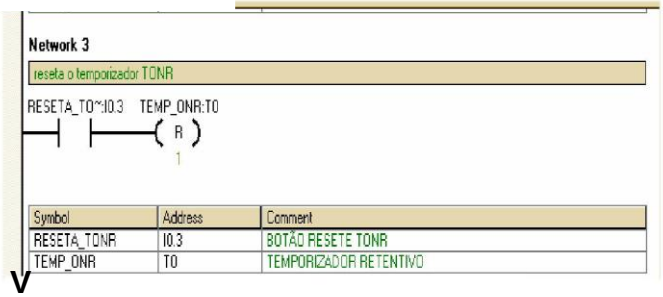


Fig. 58 - Exemplo de utilização do temporizador TONR



V

amos praticar? Objetivo: exercitar a utilização do temporizador.

Exercício – Partida Estrela Triângulo.

Elabore a rotina de programação particionada no CLP, que atenda as condições do circuito auxiliar de comando por partida estrela\triângulo, para um motor de indução trifásico.

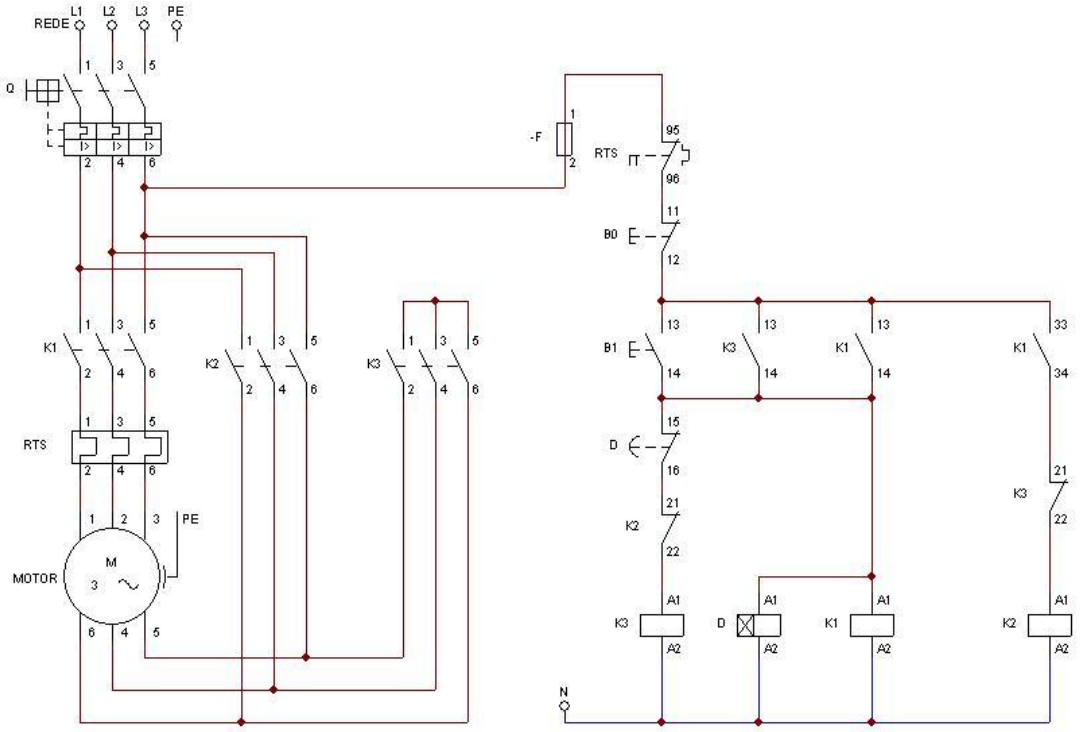
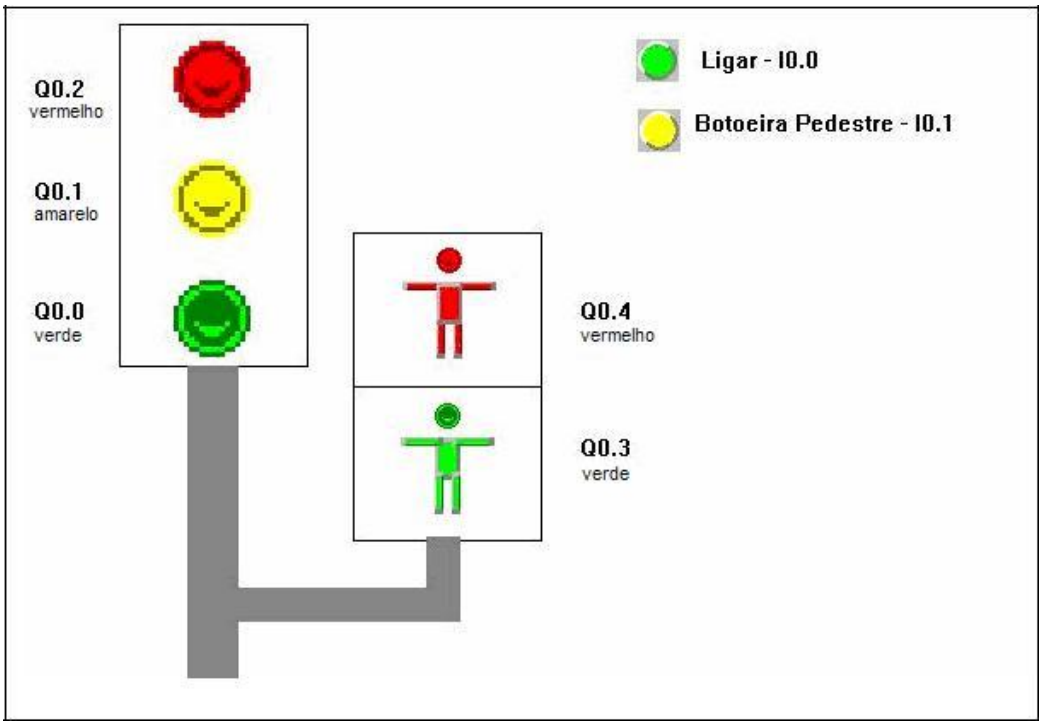


Fig. 59 – Diagrama de força e comando da partida estrela triângulo – exercício partida estrela triângulo

Vamos praticar? Objetivo: exercitar a utilização do temporizador.

Exercício – Semáforo com Botão de Pedestre

Fig. 60 – Diagrama de simulação do exercício semáforo com botão de pedestre



Elabore a rotina de programação seguindo as orientações:

O funcionamento do semáforo será iniciado pela botoeira (I0.0). Quando a mesma for pressionada, o semáforo deverá iniciar em verde;

A sinaleira verde deverá permanecer durante 40 segundos energizada;

Após 40 segundos em verde, o semáforo deverá ir para amarelo e permanecer neste estado por 5 segundos;

Após 5 segundos em amarelo, o semáforo deverá ir para vermelho e permanecer neste estado por 15 segundos;

Após 15 segundos em vermelho, o semáforo deverá voltar para verde e reiniciar seu ciclo de funcionamento;

Enquanto o semáforo estiver em verde ou amarelo a indicação do semáforo do pedestre deverá estar em vermelho;

Enquanto o semáforo estiver em vermelho a indicação do semáforo do pedestre deverá estar em verde; Se a botoeira do pedestre for pressionada o semáforo deverá ir para amarelo, desde que o verde já tenha passado 20 segundos energizado para se garantir o fluxo de veículos.

Faça um diagrama elétrico indicando as conexões no CLP dos dispositivos de entrada e saída do semáforo.

Contadores

Estas instruções estão contidas na pasta Counters. O Step 7–200 dispõe de 6 tipos de contadores: 3 para contagem de eventos de baixa velocidade e 3 para contagem de eventos de alta velocidade (High Speed). Neste material didático iremos nos referir apenas aos contadores para eventos em baixa velocidade. São eles:

- CTU – Contador Crescente;
- CTD – Contador Decrescente;
- CTUD – Contador Crescente e Decrescente.

Existem 266 endereços a serem utilizados nos contadores, que vão de C0 a C255, o valor máximo de contagem é 32.676.

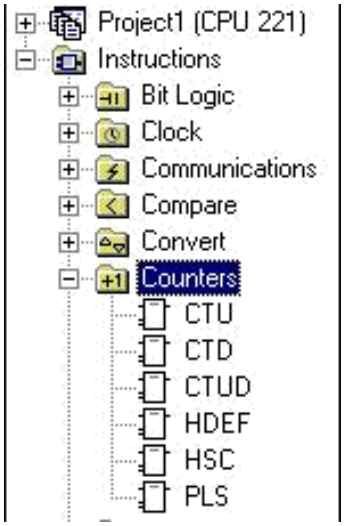


Fig. 61 – Menu Counters

Contador Crescente – CTU (Count Up)

Este contador possui uma entrada CU (Count Up) para a contagem crescente de eventos, uma entrada R (Reset) para zerar a contagem e o campo PV (Preset Value = valor prefixado) onde será inserida a quantidade de eventos a serem contados.

A cada transição de 0 para 1 na entrada CU do contador, é incrementada uma unidade na sua contagem. Quando o contador atingir o valor de contagem estabelecido em PV o seu endereço irá para o nível lógico 1, retornando para o nível lógico 0 quando for dado um pulso na entrada R (Reset).

Vamos testar? Execute a rotina, a seguir, no CLP e analise o resultado

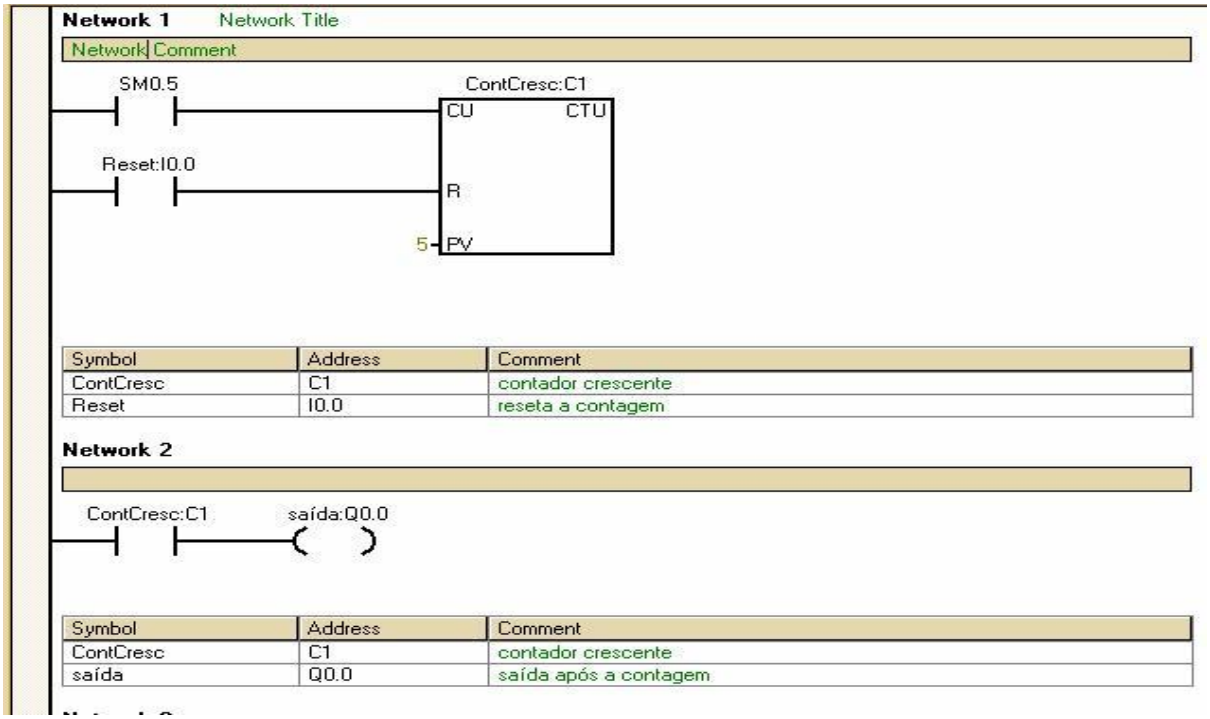


Fig. 62 – Exemplo de utilização do contador crescente

Contador Decrescente – CTD (Count Down)

Este contador possui uma entrada CD (Count Down) para a contagem decrescente de eventos, uma entrada LD (Load Input = Alimenta Entrada) para carregar a quantidade de eventos a serem contados e o campo PV onde será inserida a quantidade de eventos a serem contados.

Para iniciar a contagem deste contador, deve-se dar um pulso na entrada LD, para que o mesmo carregue o valor de contagem. Após isso, o mesmo estará habilitado para realizar a contagem de forma regressiva, desde o valor escolhido em PV até zero. A cada transição do nível lógico 0 para o nível lógico 1, na entrada CD será decrementada uma unidade no valor de contagem do contador; quando a contagem do contador zerar o seu endereço irá para o nível lógico 1.

Vamos testar? Execute a rotina, a seguir, no CLP e analise o resultado

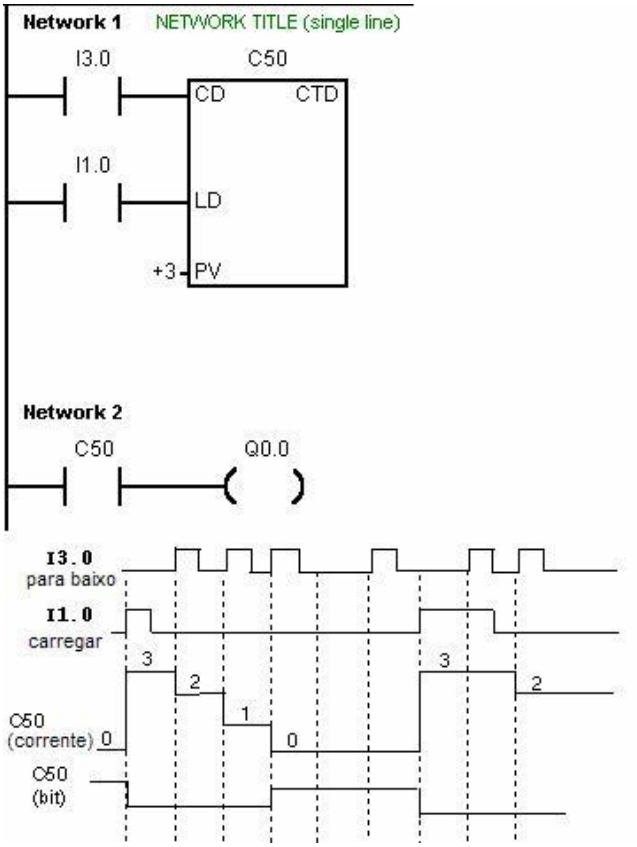


Fig. 63 – Exemplo de utilização do contador decrescente

Contador Crescente e Decrescente – CTUD (Count Up/Down)

Este contador possui uma entrada CU para a contagem crescente de eventos, uma entrada CD para contagem decrescente de eventos, uma entrada R para zerar a contagem e um campo PV onde será inserida a quantidade de

eventos a serem contados.

A cada transição de 0 para 1 na entrada CU do contador é incrementada uma unidade na sua contagem, enquanto que na entrada CD, cada transição dessa corresponderá a uma unidade decrementada na contagem do referido contador. Quando o contador atingir o valor de contagem, estabelecido em PV, o seu endereço irá para o nível lógico 1, retornando para o nível lógico 0 quando for dado um pulso na entrada R ou pulsos na entrada CD que tornem o valor de contado menor que o valor de PV.

Este contador conta eventos de –32.768 a +32.676.

Vamos testar? Execute a rotina, a seguir, no CLP e analise o resultado

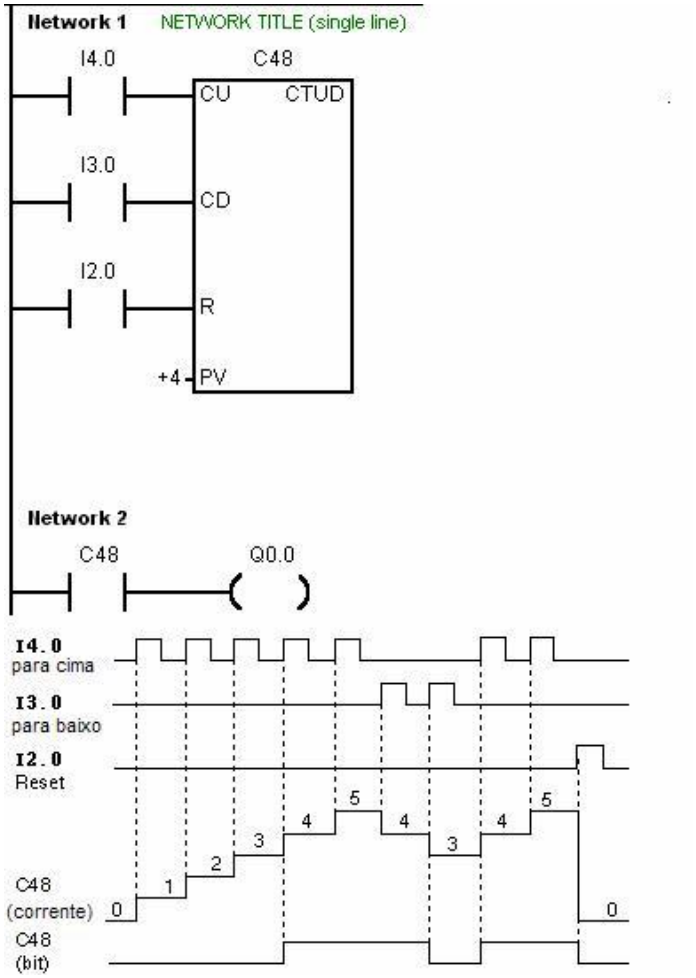


Fig. 64 - Exemplo de utilização do contador crescente - decrescente

Vamos praticar? Objetivo: exercitar a utilização do SET/RESET, do sensor digital para informação de posição e do contador. **Exercício -** Esteira Transportadora de Caixas (opção 1).

Elabore a rotina de programação seguindo as orientações:

O processo de encaixotamento será iniciado pela botoeira liga. Quando a mesma for pressionada, a esteira de transporte de caixas (Q0.0) deverá ser acionada;

O sensor S1 (I0.3) deverá interromper o funcionamento da esteira de transporte de caixas. Para que as mesmas possam ser preenchidas com os produtos, ao mesmo tempo a esteira de transporte de produtos (Q0.1) deverá ser acionada;

O sensor S2 (I0.4) será responsável pela contagem dos produtos. Cada caixa deve ser preenchida com 5 unidades do produto;

Quando a caixa estiver completamente preenchida, o funcionamento da esteira de transporte de produtos (Q0.1) deverá ser interrompido e o funcionamento da esteira de transporte de caixas deverá ser retomado, para que outra caixas possam ser preenchidas;

O processo de encaixotamento de produtos deverá ser contínuo.

Faça um diagrama elétrico indicando as conexões, no CLP, dos dispositivos de entrada e saída do processo.

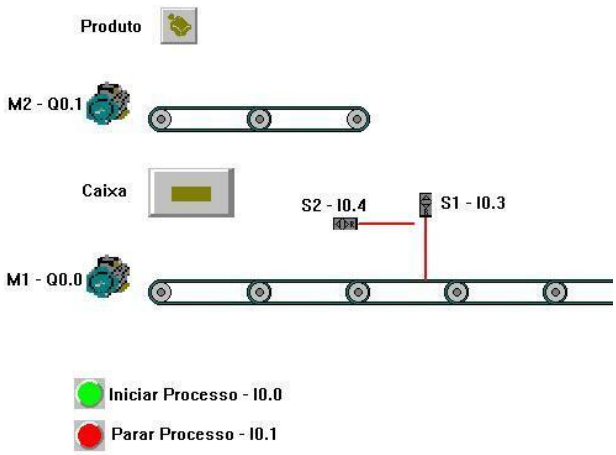


Fig. 65 – Diagrama de simulação do exercício esteira transportadora (opção 01)

Vamos praticar? Objetivo: exercitar a utilização do SET/RESET, do sensor digital para informação de posição e do contador.
Exercício - Esteira Transportadora de Caixas (opção 2).

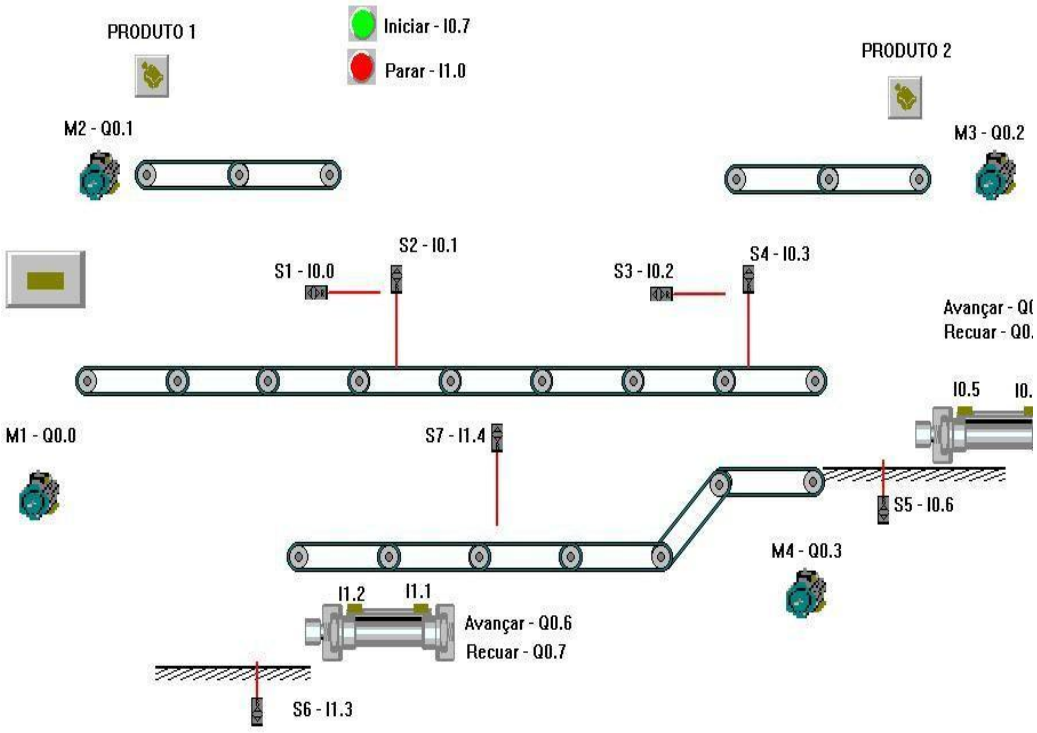


Fig. 66 – Diagrama de simulação do exercício (opção 02)

Elabore a rotina de programação seguindo as orientações:

O processo será iniciado pela botoeira iniciar. Quando a mesma for pressionada, o motor M1, responsável pelo transporte das caixas, deverá ser habilitado;

Quando a caixa atingir o sensor S2 o motor M1 deverá ser desabilitado e o motor M2, responsável pelo transporte do produto 1, deverá ser habilitado. Quando a caixa tiver com dois pacotes do produto 1, o motor M2 deverá ser desabilitado e o motor M1 deverá ser habilitado novamente;

Quando a caixa atingir o sensor S4 o motor M1 deverá ser desabilitado novamente e o motor M3, responsável pelo transporte do produto 2, deverá ser habilitado. Quando a caixa estiver com dois pacotes do produto 2, o motor M3 deverá ser desabilitado e o motor M1 deverá ser habilitado novamente, se não houver caixa no sensor S2;

Quando a caixa atingir o sensor S5 o motor M1 deverá ser desabilitado, o motor da esteira M4 deverá ser habilitado e o cilindro 1 deverá avançar para enviar a caixa para a esteira do motor M4. Quando a caixa atingir a esteira do motor M4, o cilindro 1 deverá recuar e o motor M1 deverá ser habilitado novamente, se não houver caixas nos sensores S2 e S4;

Quando a caixa atingir o sensor S6 o motor M4 deverá ser desabilitado e o cilindro 2 deverá avançar para enviar a caixa para o galpão de estocagem. Quando a caixa for enviada, o cilindro deverá recuar e aguardar a chegada de outra caixa para que possa avançar novamente;

A quantidade de caixas embaladas por dia, com a quantidade correta de produtos, deverá ser registrada; para isto utilize o sensor S7;

- O processo deverá ser contínuo;
- A qualquer momento, o processo poderá ser interrompido pressionando-se a botoeira parar, sendo retomado do mesmo ponto ao se pressionar a botoeira iniciar;
- Utilize contadores e comparadores para realizar a automação deste processo;

OBS: Deverá ser utilizada a CPU 224 no simulador do CLP.

Vamos praticar? Objetivo: exercitar a utilização do SET/RESET, do temporizador e contador.

Exercício - Carimbo Pneumático de Chapas

Fig. 67 – Diagrama de simulação do exercício carimbo pneumático de chapas

Elabore a rotina de programação seguindo as orientações:

O processo de carimbo de chapas será iniciado pela botoeira liga.

Quando a mesma for pressionada, a esteira de transporte de chapas (Q0.0) deverá ser acionada;

Quando a chapa atingir o sensor S3 (I0.4), a esteira deverá parar e o pistão do carimbo deverá avançar (Q0.1) para pressionar a chapa durante 5 segundos. Decorrido o tempo, o pistão do carimbo deverá recuar (Q0.2). O processo de carimbo deverá ser repetido 3 vezes em cada chapa;

Após a chapa ter sido carimbada por 3 vezes o pistão do carimbo deverá ficar recuado e a esteira deverá voltar a funcionar, retomando o processo para que as outras chapas possam ser carimbadas;

A qualquer momento o processo de carimbo das chapas poderá ser interrompido pressionando-se a botoeira desliga e retomado do mesmo ponto, ao se pressionar a botoeira liga;

O processo de carimbo das chapas deverá ser contínuo.

· **Comparadores**

Estas instruções estão contidas na pasta Compare.

O Step 7–200 dispõe de comparadores de igualdade, diferença, maior ou igual, menor ou igual, maior que e menor que. Poderemos comparar os valores dos seguintes formatos de dados: bytes, inteiros (word), duplo – inteiros (double word) e números reais. Quando a condição de comparação for alcançada, o contato do comparador irá para o nível lógico 1.

A seguir alguns exemplos:

Comparação de “igualdade” entre um byte e um número in 255)

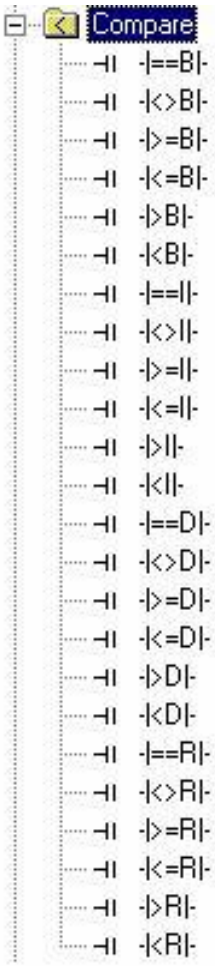


Fig. 68 – Menu Compare

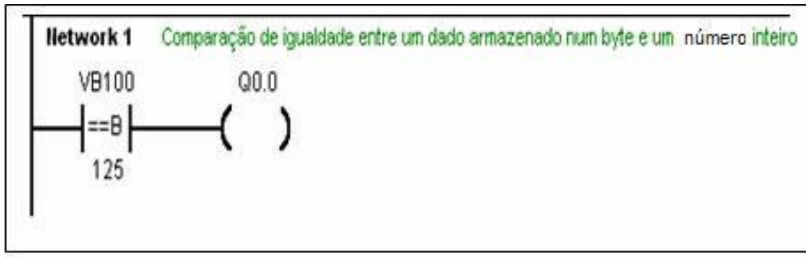


Fig. 69 - Exemplo de utilização de um comparador

A saída Q0.0 irá para o nível lógico 1, quando o valor armazenado no byte VB 100 for exatamente igual a 125.

Comparação de “maior ou igual” entre duas words

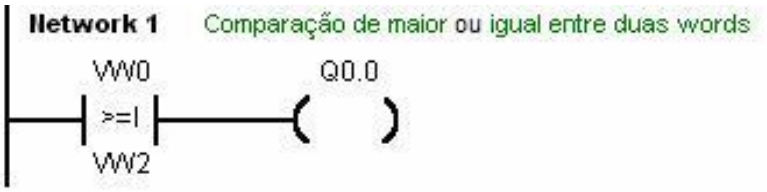


Fig. 70 - Exemplo de utilização de um comparador

A saída Q0.0 irá para o nível lógico 1, quando o valor armazenado na Word VW0 for maior ou igual ao valor armazenado na Word VW2.

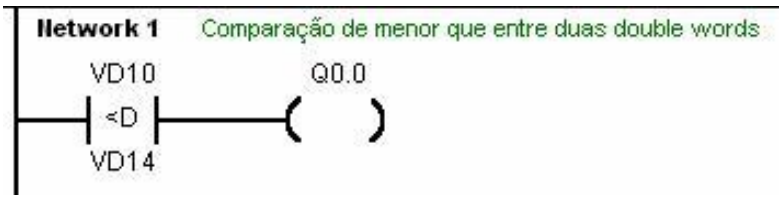
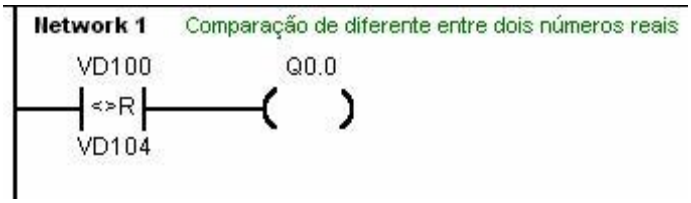


Fig. 71 – Exemplo de utilização de um comparador

Comparação de “menor que” entre duas double words



ndo o valor armazenado na
do na Double Word VD14.
meros reais

A saída Q0.0 irá para o nível lógico 1 sempre que os números reais armazenados nas Double Words VD100 e VD104 forem diferentes.

Vamos testar? Execute a rotina a seguir no CLP e analise o resultado

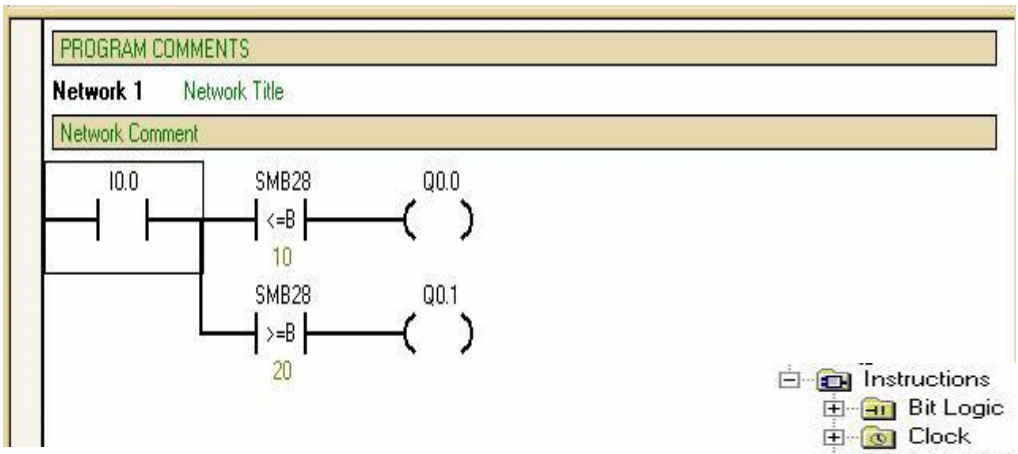


Fig. 72 – Exemplo de utilização de comparador

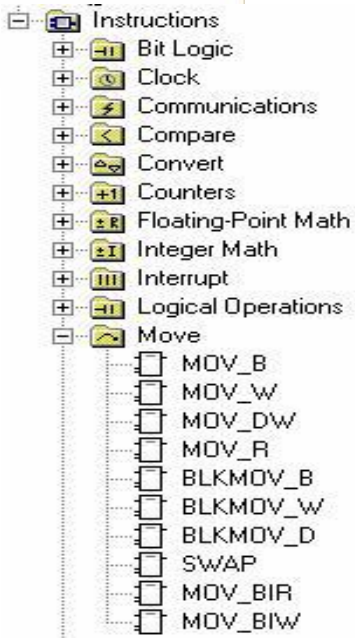


Fig. 73 – Menu Move

- **Blocos de movimentação de dados**

Estas instruções estão contidas na pasta Move.

O Step 7–200 dispõe de 3 tipos de ferramentas para a movimentação de dados.

- Move
- Block Move
- Swap

Essas ferramentas têm como função transferir o conteúdo que está alocado em uma certa região de memória para outra área de memória determinada pelo usuário.

Move (mover)

A instrução MOVE, sempre que for habilitada na sua entrada EN, moverá o dado armazenado no campo IN para uma área de memória determinada pelo usuário em OUT.

É a partir do bloco move que realizamos o controle das saídas analógicas do S7-200.

O dado de entrada pode ser uma constante M, V, I, O, AC ou SM, no

formato de byte, word ou double word. O dado de saída deverá ser obrigatoriamente no mesmo formato do dado de entrada.

Vamos testar? Execute a rotina a seguir no CLP e analise o resultado

Fig. 74 – Exemplo de utilização do Move

Toda vez que a entrada (EN - Enable IN = habilita entrada) estiver no nível lógico 1, a instrução MOVE será habilitada movendo o dado da entrada analógica AIW0 (campo IN) para a saída analógica AQW0 (campo OUT).

IN – Endereço de Origem;

OUT – Endereço de Destino.

ENO – (Enable Out = habilita saída).
Se a instrução for executada corretamente, teremos nível lógico 1 nesta saída, caso contrário, havendo algum erro na execução da instrução, teremos nível lógico igual a ZERO. Esta saída poderá ser usada para sinalizar a execução correta ou não da instrução.

Block Move (mover blocos)

A instrução BLOCK MOVE, sempre que for habilitada na sua entrada EN, moverá a quantidade de endereços consecutivos N, a partir do endereço inicial no campo IN para outra área de memória determinada pelo usuário no campo OUT.

O dado de entrada pode ser uma constante M, V, I, O, AC ou SM, no formato de byte, word ou double word. O dado de saída deverá ser obrigatoriamente no mesmo formato do dado de entrada.

Vamos testar? Execute a rotina, a seguir, no CLP e analise o resultado

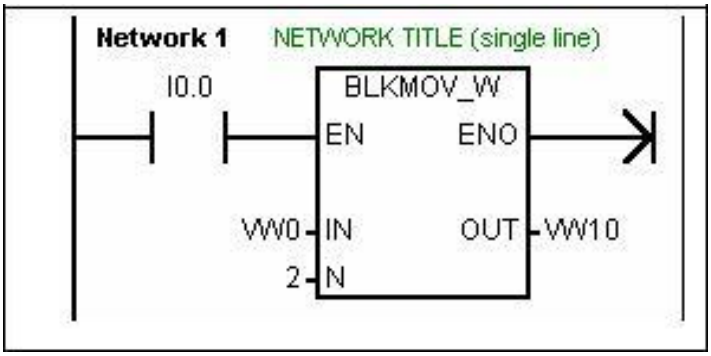


Fig. 75 - Exemplo de utilização do Block Move

Toda vez que a entrada EN estiver no nível lógico 1, a instrução BLOCK MOVE será habilitada movendo os dados armazenados nos N endereços estipulados para a outra área de memória definida em OUT. No caso do exemplo anterior, os dados armazenados na área de memória VW0 e VW2 serão movidos para as áreas de memória VW10 e VW12.

IN – Endereço Inicial;

N – Quantidade de endereços a serem movidos a partir do inicial;

OUT – Endereço inicial de destino.

Swap (trocar)

Esta é uma instrução especial onde são movidos os bytes internos de uma word, da seguinte forma:

Toda vez que a entrada EN estiver no nível lógico 1, a instrução ficará invertendo o byte mais significativo, com o byte menos significativo, até que a entrada EN volte para o nível lógico 0.

Se temos em VW100 = D6 C3, depois do SWAP, teremos em VW100 = C3 D6.

Vamos testar? Execute a rotina, a seguir, no CLP e analise o resultado

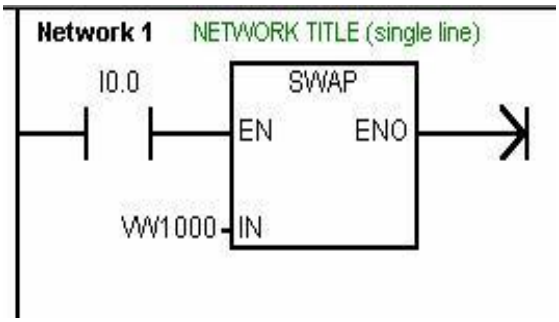


Fig. 76 – Exemplo de utilização do SWAP

- **Operações Matemáticas**

Estas instruções estão contidas nas pastas Floating-Point Math e Integer Math. As CPU do S7-200 possuem todas as operações matemáticas básicas (adição, subtração, multiplicação e divisão) em seu Set de instruções. Algumas CPU, além das operações básicas, também possuem operações do tipo: seno, co-seno, tangente, raiz quadrada, exponencial, etc.

Estas operações podem ser feitas em formato de Inteiro (I), Duplo Inteiro (DI) e Real (R). Para que se possa executar essas operações, faz-se necessário que as duas grandezas que serão operadas estejam no mesmo formato (INT / INT, DINT / DINT, REAL / REAL). Caso as duas grandezas não estejam no mesmo formato, é necessário o uso de operações de conversão.

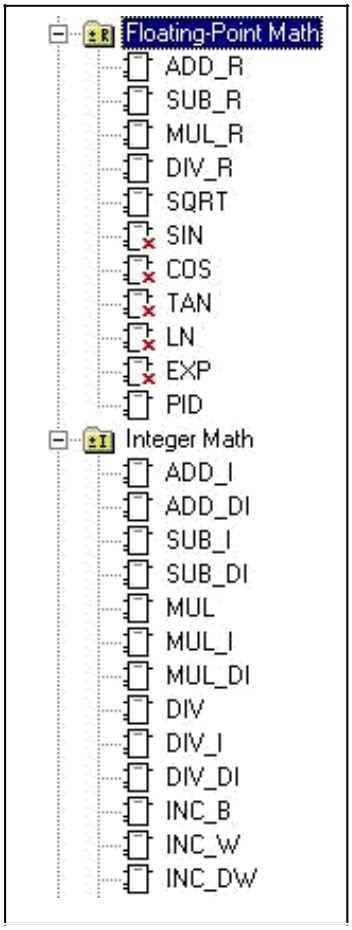


Fig. 77 – Menu Integer Math

Adição de dois valores inteiros (16 bits) – ADD_I (addition Integer)

Sempre que a entrada EN estiver no nível lógico 1 as entradas IN1 e IN2 serão somadas e o resultado da soma será guardado na área de memória estabelecida em OUT.

A soma não pode ultrapassar 32.767, valor máximo para armazenamento em uma word.

Vamos testar? Execute a rotina, a seguir, no CLP e analise o resultado

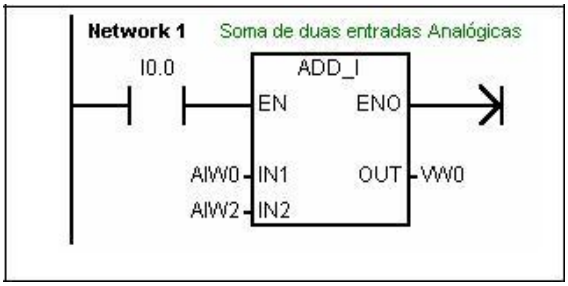


Fig. 78 – Exemplo de utilização do ADD

Subtração de dois valores reais (32 bits) - SUB_R (subtract real)

Sempre que a entrada EN estiver no nível lógico 1, as entradas IN1 e IN2 serão subtraídas e o resultado da subtração será guardado na área de memória estabelecida em OUT.

Obs: todo número real deve ser armazenado no formato double word, em função da casa decimal.

Vamos testar? Execute a rotina a seguir no CLP e analise o resultado

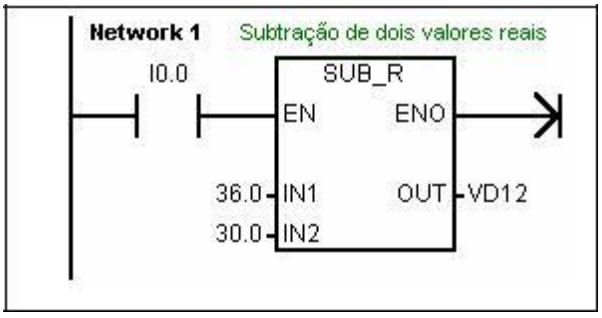


Fig. 79 - Exemplo de utilização do SUB_R

Multiplicação de dois valores inteiros de 16 bits, gerando um inteiro de 32 bits (duplo inteiro) – MUL (multiply integer to double integer)

Sempre que a entrada EN estiver no nível lógico 1, as entradas IN1 e IN2 serão multiplicadas e o resultado da multiplicação será guardado na área de memória estabelecido em OUT.

Obs: neste caso, como a multiplicação pode ultrapassar o valor numérico máximo que pode ser armazenado numa Word que é 32.767, devemos então enviar o resultado para uma área de memória maior, no caso uma double word.

Vamos testar? Execute a rotina, a seguir, no CLP e analise o resultado

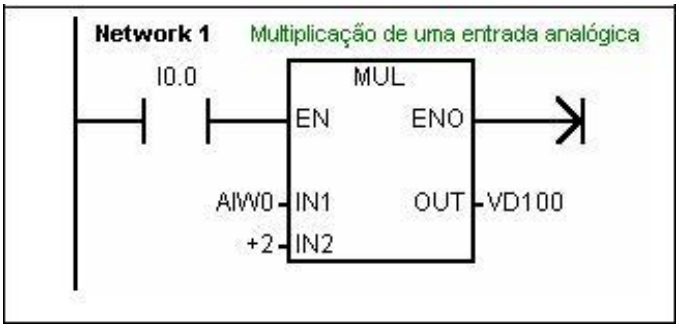


Fig. 80 - Exemplo de utilização do MUL

Conversores

Estas instruções estão contidas na pasta Convert.

O Step 7–200 dispõe de algumas ferramentas para conversão de dados de um formato para outro.

TRUNC: (Truncate – truncar, cortar parte)
Converte um dado no formato real para duplo inteiro. Só a parte inteira do número real é convertida, a fração é descartada.
ROUND: (Round – arredondar) Converte um dado no formato real para duplo inteiro. Se a fração for 0,5 ou maior, o arredondamento será para mais.

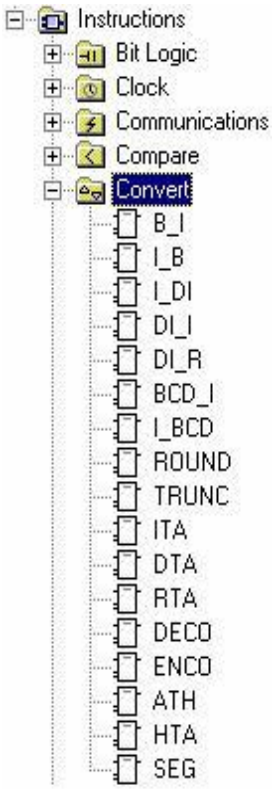


Fig. 81 –

BCD_I: (BCD to Integer - binary-coded decimal to integer – código decimal binário para inteiro) converte um dado no formato BCD para inteiro; **I_BCD** faz o inverso;
DI_R: (Double Integer to Real) converte um dado no formato duplo inteiro para real; **R_DI** faz o inverso;
DI_I: (Double Integer to Integer) converte um dado no formato duplo inteiro para inteiro; **I_DI** faz o inverso;
B_I: (Byte to Integer) converte um dado no formato de byte para inteiro; **I_B** faz o inverso;

Vamos testar? Execute a rotina, a seguir, no CLP e analise o resultado

Na situação a seguir, sempre que a entrada I0.0 (EM) estiver no nível lógico 1, o valor armazenado na área de memória VW0 (IN), que está no formato inteiro (16 bits), será convertido para o formato de duplo inteiro (32 bits) e armazenado no endereço VD2 (OUT).

A saída ENO do bloco irá para o nível lógico zero, caso ocorra algum erro na conversão dos dados.

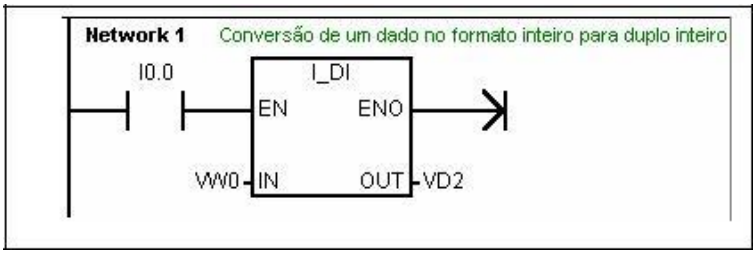


Fig. 82 – Exemplo de utilização de I_DI

- **Blocos para desenvolvimento de sub-rotinas**

Sub-rotina (SBR) é uma rotina que pode ser acessada várias vezes, durante o processamneto, mas com parâmetros físicos de acesso diferentes (variáveis globais). Isto permite que um bloco criado na sub-rotina seja utilizado diversas vezes dentro de um programa, diminuindo o tempo de desenvolvimento do programa e a memória ocupada na CPU.

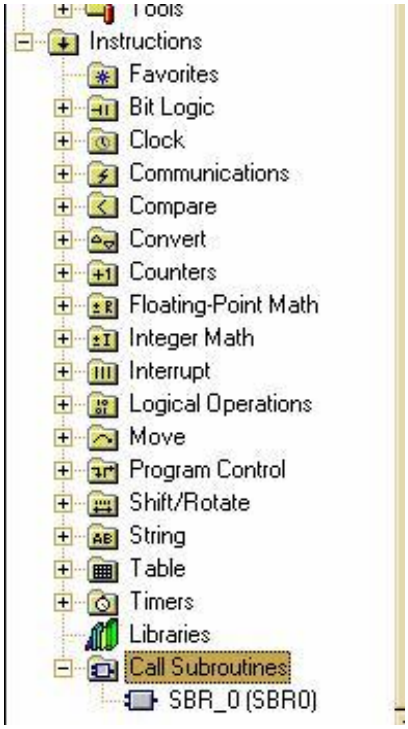


Fig. 83 – Menu Call Subroutines

Dentro da sub-rotina criamos a lógica em Ladder, em seguida preenchemos a tabela de variáveis locais e, por fim, associamos a tabela à lógica Ladder.

Esta associação é feita digitando na lógica Ladder os mnemônicos correspondentes (symbol) usados na tabela de variáveis locais. Assim que isto é feito aparece antes do mnemônico um sinal de cerquilha (#), que caracteriza uma variável local.

Vejamos abaixo algumas variáveis:

· Endereçamento Local (End.Local):

Endereço relativo da memória local criado automaticamente pelo sistema.

· Nome (Symbol):

Nome simbólico da variável, mnemônico. Este nome será usado na lógica do programa.

· Tipo da variável (Var. Type):

IN – parâmetro de entrada OUT

– parâmetro de saída

IN/OUT – parâmetro de entrada e saída

TEMP – são variáveis válidas exclusivamente no bloco em que foram definidas.

· Tipo do dado (Data Type)

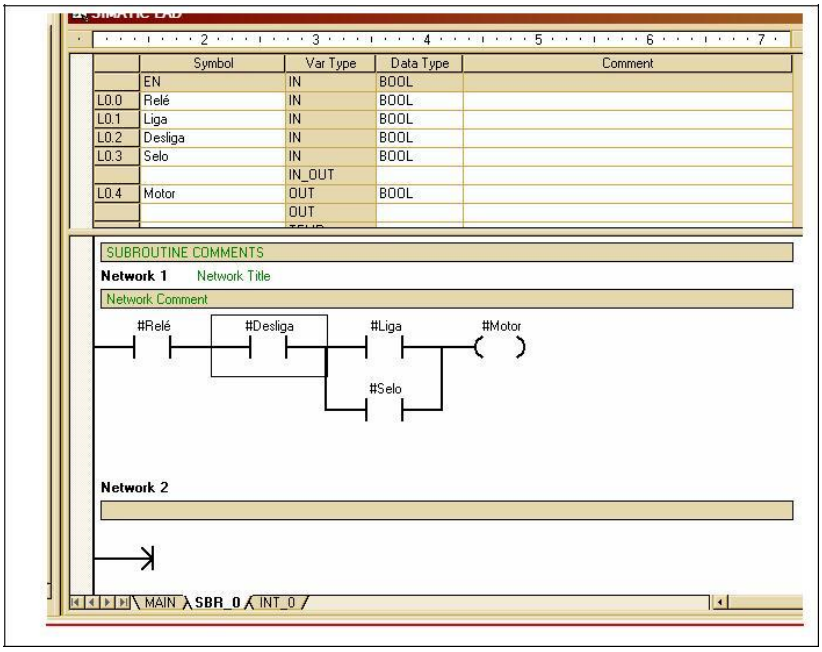
Bool; Int; Word; etc.

 Comentário (Comment):

Descritivo opcional sobre a variável

Vamos testar? Execute a rotina, a seguir, no CLP e analise o resultado

Fig. 84 – Exemplo de utilização de sub-rotina



Atenção para a barra de status. Estamos na SBR0

Obs.: Ao montar a tabela de variáveis locais, os mnemônicos usados pelo programa STEP7 – Micro/Win não serão aceitos na coluna "symbol". Se isto for feito por engano, quando tentarmos fazer o endereçamento, o símbolo de cerquilha (#) não aparecerá antes do mnemônico.

Network 1
Atenção para a barra de status. Estamos no OB1 Abaixo, à esquerda, “ call subroutines – SBR_0”

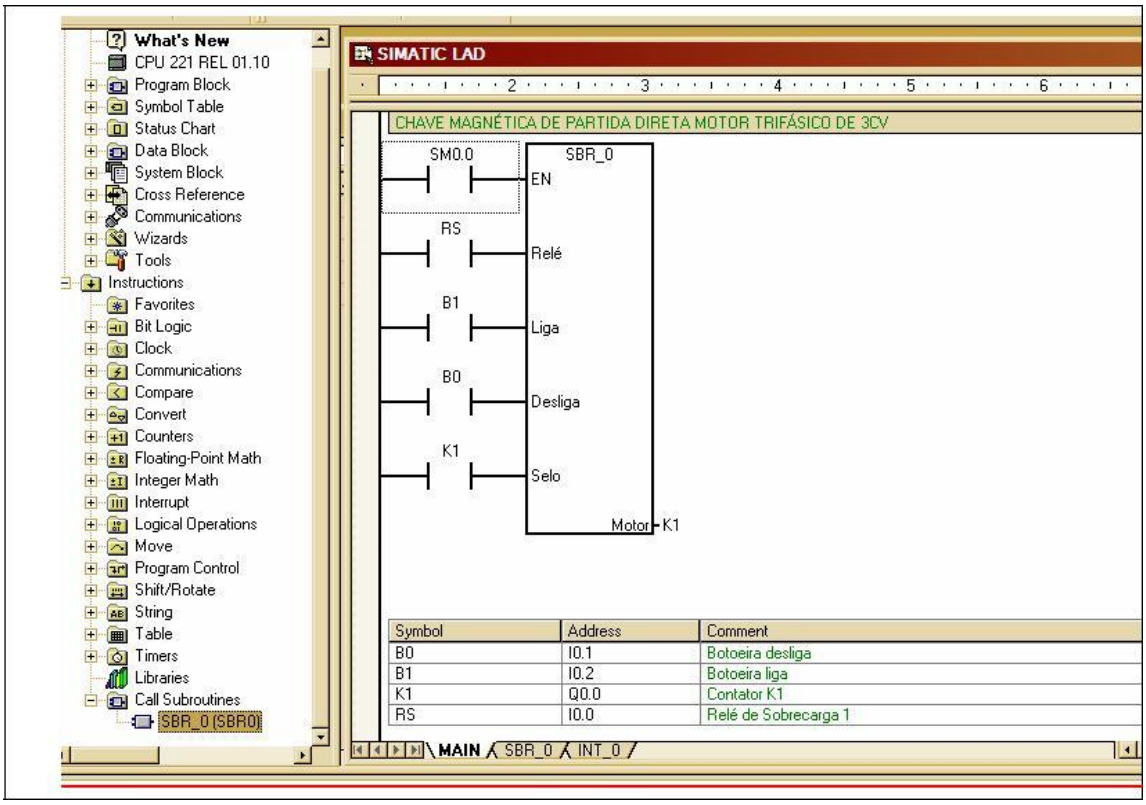


Fig. 85 – Exemplo de utilização de sub-rotina

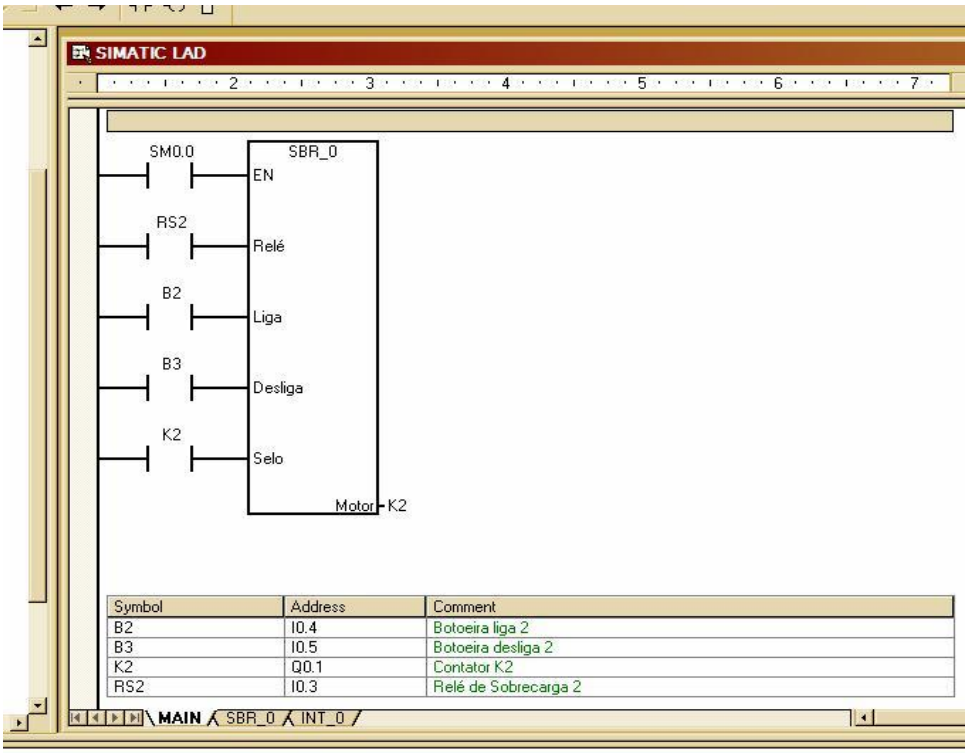


Fig. 86 - Exemplo de utilização de sub-rotina

Network 2
A mesma sub-rotina, porém com endereços físicos diferentes

Observe acima duas networks usando a mesma sub-rotina, claro que com endereços físicos (variáveis globais) diferentes, porém dentro da sub-rotina as variáveis locais são as mesmas.

Sub-rotina de Interrupção (INT) é uma rotina desenvolvida para ser acessada por um determinado evento. A ocorrência deste evento fará a CPU desviar seu processamento cíclico para executar a rotina de interrupção.

Os eventos que podem gerar o desvio de processamento da CPU estão pré-definidos na própria CPU.

A instrução “ATCH/ENI” inserida no programa determinará o desvio no momento do evento de interrupção.

MÓDULOS DE EXPANSÃO ANALÓGICOS

Sinal Analógico é a representação de uma grandeza que pode assumir, no decorrer do tempo, qualquer valor entre dois limites determinados. As grandezas analógicas elétricas tratadas por um CLP, normalmente são tensão e corrente.

Tensão: 0 a 10VCC; 0 A 5 VCC; -5 a +5VCC; -10 a +10VCC.

Corrente: 0 a 20mA; 4 a 20mA

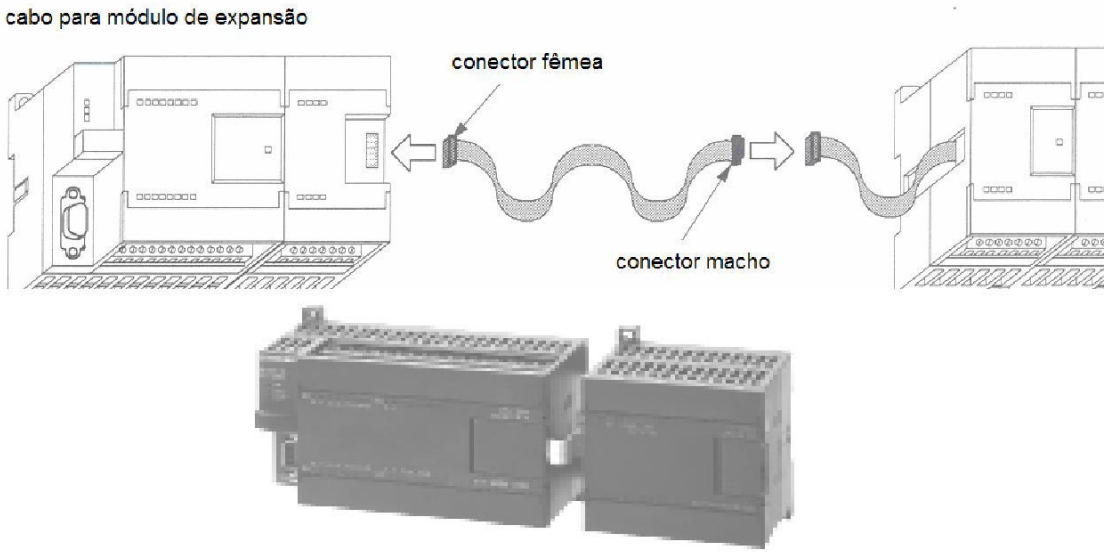


Fig. 87 - CPU SIEMENS S7-200 com Módulo de Expansão

Revisão

Representação Binária: os números binários são representados por dígitos que recebem denominações específicas em função de sua utilização.

- **Bit:** Qualquer dígito de um número binário é um “bit” (binary digit). Exemplo: 1010, este número é formado por 4 dígitos, ou seja, 4 bits.
- **Byte:** A associação de 8 bits forma um “byte” (binary term). Exemplo: 1110 1100 (8 bits = 1 byte)
- **Word:** Número binário formado por dois bytes. Exemplo: 1010 1110

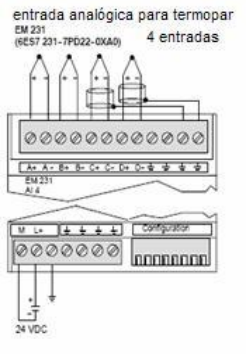


Fig. 88 – Módulo analógico

(1ë byte) 0110 1100 (2ë byte)

- **Double Word:** Número binário formado por duas words. Exemplo:

(2ë Word).

O sinal analógico pode ser:
Unipolar: por exemplo 0 a 50mV
Bipolar: por exemplo +/- 25mV

Existem dois módulos analógicos:
EM 231 e EM 235

Entrada Analógica

Recebe sinal analógico e converte em valores numéricos. Os principais dispositivos utilizados nas entradas analógicas são:

1. sensores de pressão manométrica;
2. sensores de pressão mecânica (strain gauges – células de carga);
3. taco geradores;
4. transmissores de temperatura;

Saída Analógica

Converte valores numéricos em sinais de tensão ou corrente, em geral 0 a 10VCC ou 0 a 5VCC, e corrente de 0 a 20mA ou 4 a 20mA. Estes sinais

são utilizados para controlar dispositivos do tipo:

- válvulas proporcionais;
- motores CC;
- servo-motores CC;
- posicionadores rotativos.

Os potenciômetros de ajuste analógico ficam situados sob a tampa frontal do módulo S7 -200. Esses potenciômetros podem ser ajustados para aumentar ou diminuir valores que são armazenados em “Bytes de Memória Especial” (SMB28 e SMB29).

Estes valores, só de leitura, podem ser usados pelo programa para uma variedade de funções, como atualizar o valor atual para um temporizador ou contador, carregar ou mudar os valores prefixados ou fixar limites.

SMB28 mantém o valor digital que representa a posição 0 do ajuste analógico.
SMB29 mantém o valor digital que representa a posição 1 do ajuste analógico.

Os ajustes serão feitos por uma chave de fenda pequena. Gire o potenciômetro à direita para aumentar o valor, e à esquerda para diminuí-lo.

Vamos praticar? Objetivo: exercitar a utilização do temporizador e do sensor analógico para informação de nível.

Exercício - Controle de Nível com Sensor Analógico

fig. 89 – Diagrama de simulação do exercício sensor analógico

Elabore a rotina de programação seguindo as orientações:

Obs: O sensor de nível analógico, mede de 0 – 100.000 litros, numa escala de 0 – 10 V;

- O processo será iniciado pela botoeira Ligar. Quando a mesma for pressionada, a eletroválvula de entrada Q0.0 deverá ser habilitada;
- Quando o reservatório atingir seu nível médio a eletroválvula de entrada deverá ser desabilitada e o sinalizador Q0.3 deverá ser habilitado;
- Após a eletroválvula de entrada ser desabilitada, devemos contar 10 segundos para que a mesma possa ser habilitada novamente. Quando a eletroválvula de entrada for habilitada novamente, o sinalizador Q0.3 deverá ser desabilitado, pois, o tanque não se encontra mais no seu nível médio;
- Quando o reservatório atingir seu nível máximo (cheio) a eletroválvula de entrada deverá ser desabilitada novamente e o sinalizador Q0.2 deverá ser habilitado;
- Após a eletroválvula de entrada ser desabilitada, o temporizador conta 10 segundos para que a eletroválvula de saída seja habilitada e o sinalizador Q0.2 seja desabilitado, pois o tanque não se encontra mais no seu nível máximo (cheio);
- Quando o tanque estiver completamente vazio o sinalizador Q0.4 deverá ser habilitado;
- O processo não é contínuo. Para ser reiniciado, a botoeira liga deverá ser pressionada novamente;
- A qualquer momento, o processo de enchimento poderá ser interrompido pressionando-se a botoeira parar, e retomado do mesmo ponto ao se pressionar a botoeira iniciar.

OBS: Deverá ser utilizada a CPU 222 e o Módulo de Expansão EM235 no simulador do CLP.

Vamos praticar? Objetivo: exercitar a utilização do temporizador e dos sensores digitais para informação de nível e de posição.

Exercício – Controle de Enchimento de Silo com Sensores de Nível.

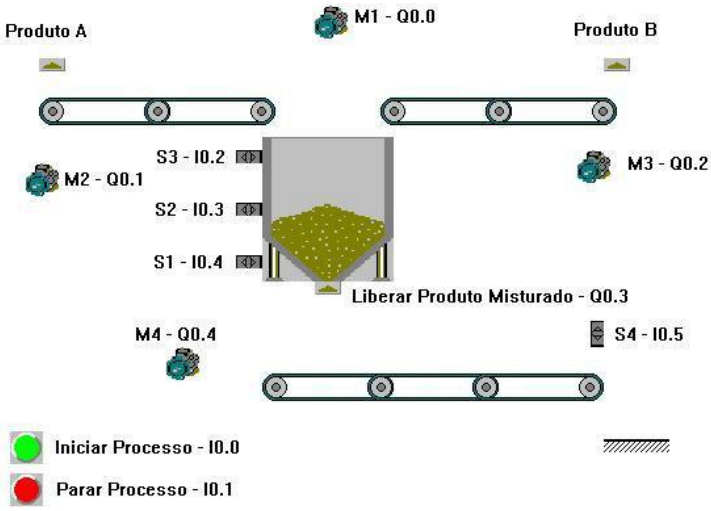


Fig. 90 – Diagrama de simulação do exercício controle de enchimento de silo com sensores de nível

Elabore a rotina de programação seguindo as orientações:

- O enchimento do silo será iniciado pela botoeira liga;
- A qualquer momento, o processo de enchimento do silo poderá ser interrompido pressionando-se a botoeira desliga, e retomado do mesmo ponto, ao se pressionar a botoeira liga;
- A esteira acionada por M2 deve injetar o produto A até a metade do silo;
- A esteira acionada por M3 deve completar o enchimento do silo com o produto B;
- O controle de nível do silo será realizado pelos sensores S3 (nível máximo), S2 (nível médio) e S1(nível mínimo);
- Quando o silo estiver no nível máximo, o motor M1 deverá ser acionado para misturar os produtos durante 20 segundos;
- Após os produtos serem misturados, a saída (Q0.3), que libera o produto misturado, deverá ser habilitada para enviar a mistura à etapa de empacotamento. Este transporte será realizado pela esteira acionada por M4;
- A esteira acionada por M4 só deverá ser desligada quando não houver mais produto na mesma;
- O processo de enchimento do silo deverá ser contínuo.

Objetivo: exercitar a utilização do temporizador e dos sensores digitais para informação de nível.

Exercício – Controle de Enchimento de Tanque com Sensores e Motor Misturador

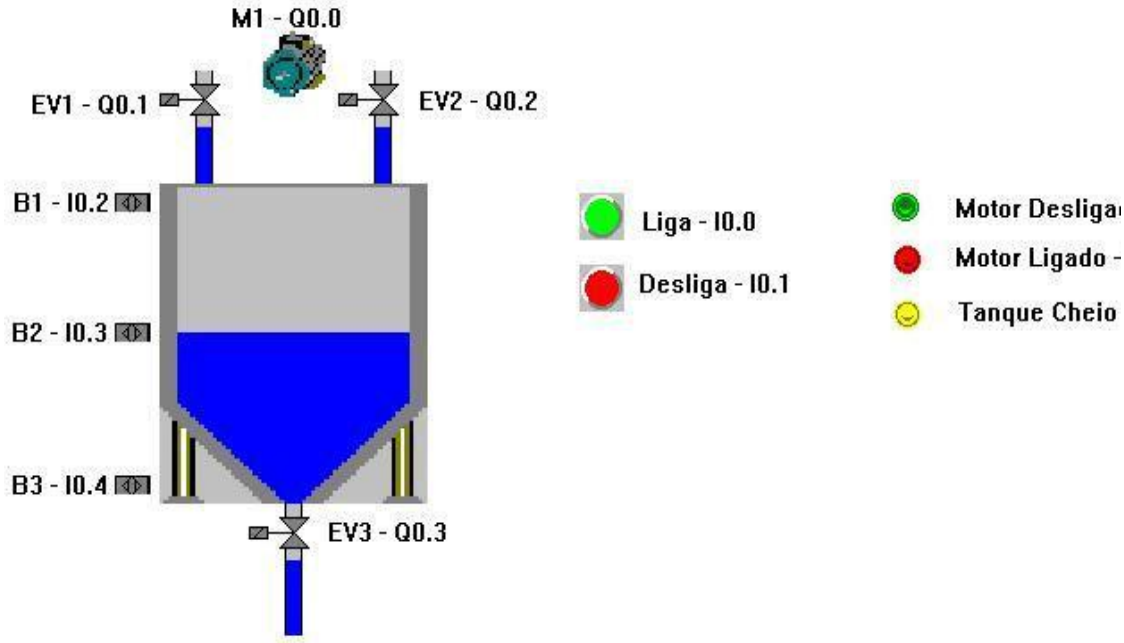


Fig. 91 – Diagrama de simulação do exercício controle de enchimento tanque com sensores de motor

Elabore a rotina de programação segundo as orientações a seguir:

- O enchimento do tanque será iniciado pela botoeira liga;
- A qualquer momento, o processo de enchimento e esvaziamento do tanque poderá ser interrompido pressionando-se a botoeira desliga e retomado do mesmo ponto, ao se pressionar a botoeira liga;
- A eletroválvula - EV1 deve injetar o produto A até a metade do tanque;
- A eletroválvula – EV2 deve completar o enchimento do tanque com o produto B;
- O controle de nível do tanque será realizado pelos sensores B1 (nível máximo), B2 (nível médio) e B3 (nível mínimo);
- Quando o tanque estiver no nível máximo, o motor M1 deverá ser acionado para misturar os produtos durante 10 segundos;
- Após os produtos serem misturados a eletroválvula – EV3 deverá ser habilitada para enviar a mistura à etapa de envasamento;
- Só depois que toda a mistura tiver sido enviada para a etapa de envasamento, aí então o tanque poderá ser cheio novamente.
- Sinalize o funcionamento do motor e o enchimento do tanque.

Vamos praticar? Objetivo: exercitar a utilização dos sensores digitais para informação de nível.

Exercício – Enchimento de Tanque com Motobomba

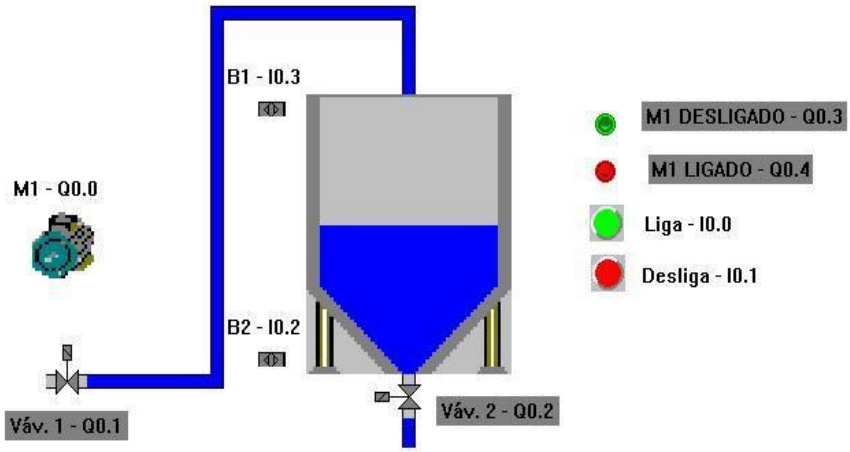


Fig. 92 – Diagrama de simulação do exercício enchimento de tanque com motobomba

Elabore a rotina de programação seguindo as orientações:

- O enchimento do tanque será iniciado pela botoeira liga, porém a motobomba de enchimento do tanque só deverá ser habilitada se o tanque não estiver no nível máximo;
- A qualquer momento, o processo de enchimento do tanque poderá ser interrompido pressionando-se a botoeira desliga;
- Ao mesmo tempo em que a motobomba é acionada a válvula 1 deverá ser habilitada, para evitar que a motobomba funcione em vazio;
- O controle de nível do tanque será realizado pelos sensores B1 (nível máximo) e B2 (nível mínimo);
- Quando o tanque estiver no nível máximo, a válvula 2 deverá ser habilitada para que o produto possa ser enviado para a outra etapa do processo;
- Só depois que todo o produto tiver sido enviado para a outra etapa do processo, aí então o tanque poderá ser cheio novamente.
- Sinalize o funcionamento do motor.

Vamos praticar? Objetivo: exercitar a utilização dos sensores digitais para informação de nível e de posição. **Exercício –** Processo de Envasamento – opção 1

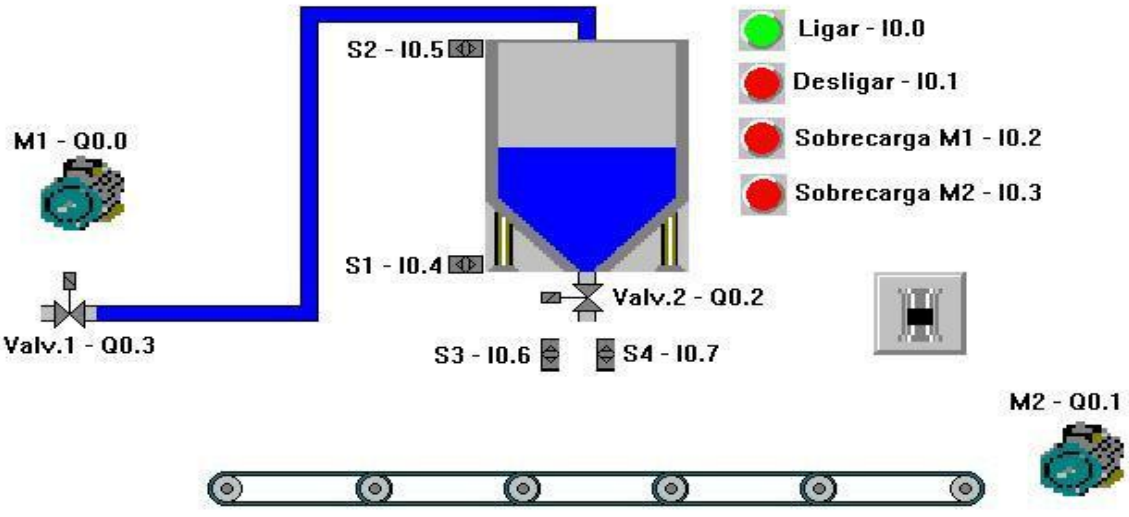


Fig. 93 – Diagrama de simulação do exercício processo de envasamento (opção 01)

Elabore a rotina de programação seguindo as orientações:

- O enchimento do tanque será realizado pelos sensores S1(nível mínimo) e S2(nível máximo), independentemente das botoeiras liga e desliga;
- A botoeira liga deverá acionar o motor M2 que é responsável pelo transporte dos vasilhames;
- A qualquer momento, o processo de envasamento do produto poderá ser interrompido, pressionando-se a botoeira desliga e retomado do mesmo ponto, ao se pressionar a botoeira liga;
- Quando o sensor S3 for habilitado, o mesmo deverá interromper o funcionamento da esteira e habilitar a válvula de enchimento, para que o vasilhame possa ser cheio com o produto;
- O sensor S4 é habilitado quando o vasilhame estiver com a quantidade correta do produto; o mesmo deverá desabilitar a válvula de enchimento e reabilitar a esteira;
- As entradas I0.2 e I0.3 representam os relés de sobrecarga dos motores M1 e M2, respectivamente;
- O processo deverá ser contínuo.

Vamos praticar? Objetivo: exercitar a utilização do temporizador e dos sensores digitais para informação de nível e de posição. **Exercício –** Processo de Envasamento – opção 2

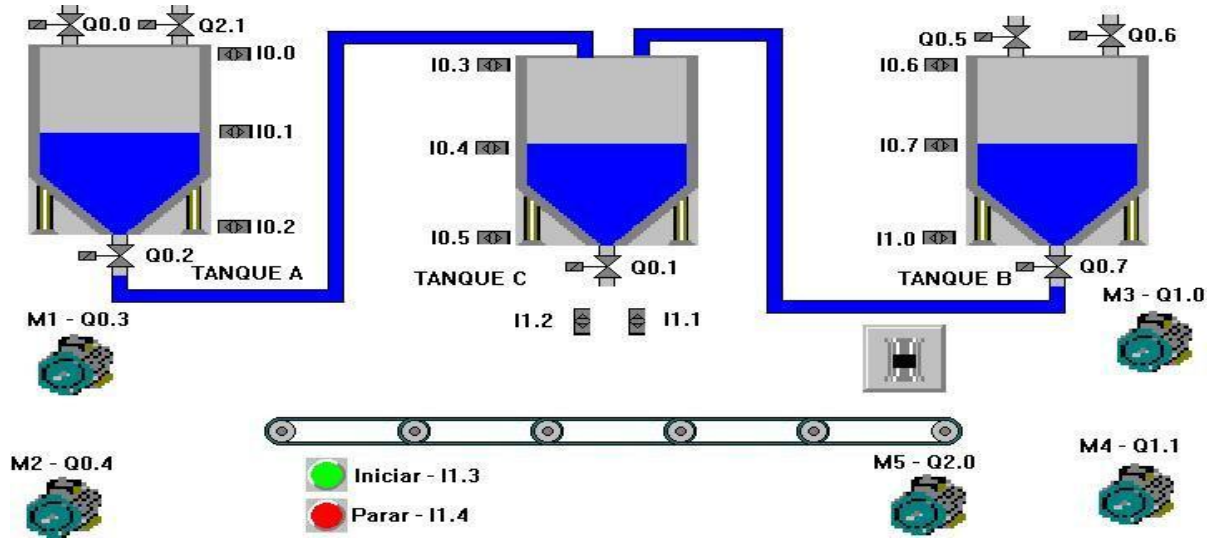


Fig. 94 – Diagrama de simulação do exercício processo de envasamento (opção 02)

Elabore a rotina de programação seguindo as orientações:

- O processo de fabricação e envasamento do produto é iniciado pela botoeira Iniciar (I1.3); ao ser pressionada, a mesma deverá iniciar o enchimento dos tanques A e B habilitando as válvulas Q0.0 e Q0.5;

- Quando os tanques A e B estiverem com nível médio, as válvulas Q0.0 e Q0.5 devem ser desabilitadas, e as válvulas Q2.1 e Q0.6 devem ser habilitadas para completar o enchimento dos respectivos tanques;
- Quando os tanques A e B estiverem cheios, os misturadores Q0.4 tanque A e Q1.1 tanque B, devem ser habilitados para misturar os produtos dos respectivos tanques durante 10 segundos;
- Após os produtos (de cada tanque) terem sido misturados, a bomba Q0.4 do tanque A deve ser habilitada, para enviar o produto do tanque A para o tanque C. O tanque C deverá receber o produto do tanque A até o seu nível médio;
- Após o tanque C ter atingido seu nível médio com o produto do tanque A, a bomba Q0.4, do tanque A, deverá ser desabilitada e a bomba Q1.0, do tanque B, deverá ser habilitada para completar o enchimento do tanque C;
- A esteira de transporte dos vasilhames deverá ser habilitada quando o tanque C estiver cheio;
- Os sensor I1.2 atua quando o vasilhame estiver no ponto correto para ser envasado, o sensor I1.1 atua quando o vasilhame estiver cheio e a válvula Q0.1 libera o produto final para encher o vasilhame;
- Quando o tanque C tiver sido esvaziado o processo de envasamento deverá ser interrompido, até que o mesmo esteja cheio novamente, com a mesma quantidade dos produtos dos tanques A e B;
- Quando o tanque C tiver novamente cheio, com a quantidade correta dos produtos dos tanques A e B, o processo de envasamento deverá ser retomado;
- A qualquer momento o processo poderá ser interrompido, pressionando-se a botoeira parar e retomado do mesmo ponto, ao se pressionar a botoeira Iniciar;
- O processo deverá ser contínuo.

OBS: Para simulação deste exercício deverá ser utilizada a CPU 224 com o módulo de expansão EM 222.

X. DISPLAY DE TEXTO - TD 200

A IHM (Interface Humano Máquina) é um equipamento que possibilita ao operador a monitoração e interação com a máquina ou processo industrial, através de um display de texto ou tela gráfica.

A IHM cumpre a função de apresentar, de forma inteligível, o status de sinais de sensores e atuadores, válvulas, motores, valores de variáveis de processo, alarmes e indicações de falhas. Através de um teclado ou tela sensível ao toque (touch screen) possibilita de forma interativa a realização de comandos, acionamento de atuadores, alterações de set points (valores ideais determinados pelo operador), mudança de manual para automático e definição de limites de funcionamento.

No teclado, a quantidade de teclas depende da capacidade de realizar funções de cada IHM. As IHM utilizadas apenas para visualização de informações ou do tipo “touch screen”, não necessitam de teclado.
No caso da IHM TD 200, utilizada pelo S7-200, toda programação é realizada no software de programação do próprio CLP, o STEP 7-Micro/WIN.

1. A opção para programação da IHM é encontrada no menu Tools (ferramentas), opção: Text Display Wizard (mágica para exibição de texto).

STEP 7-Micro/WIN - Project1 - [SIMATIC LAD]
File Edit View PLC Debug Tools Windows Help
Instruction Wizard...
Text Display Wizard...

Fig. 95 – Menu Tools – Text Display Wizard...

2. Após selecionar a opção Text Display Wizard surgirá a tela na qual deveremos selecionar NEXT (próximo).
3. Na tela a seguir selecionar “TD 200 Version 2.1 and earlier” (versão 2.1 e mais recente) e em seguida Next.

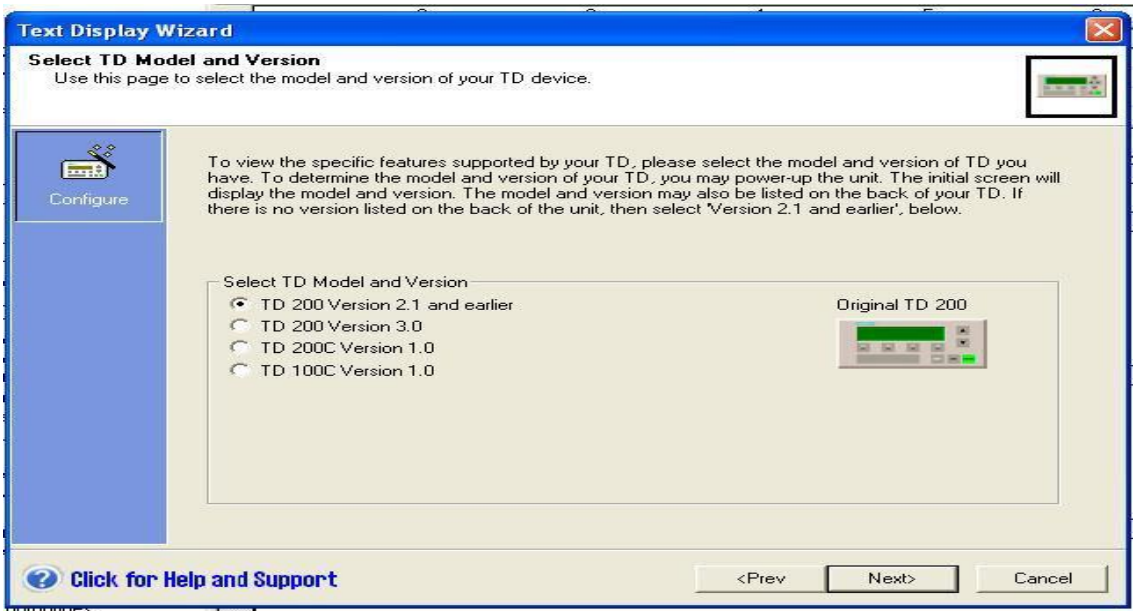
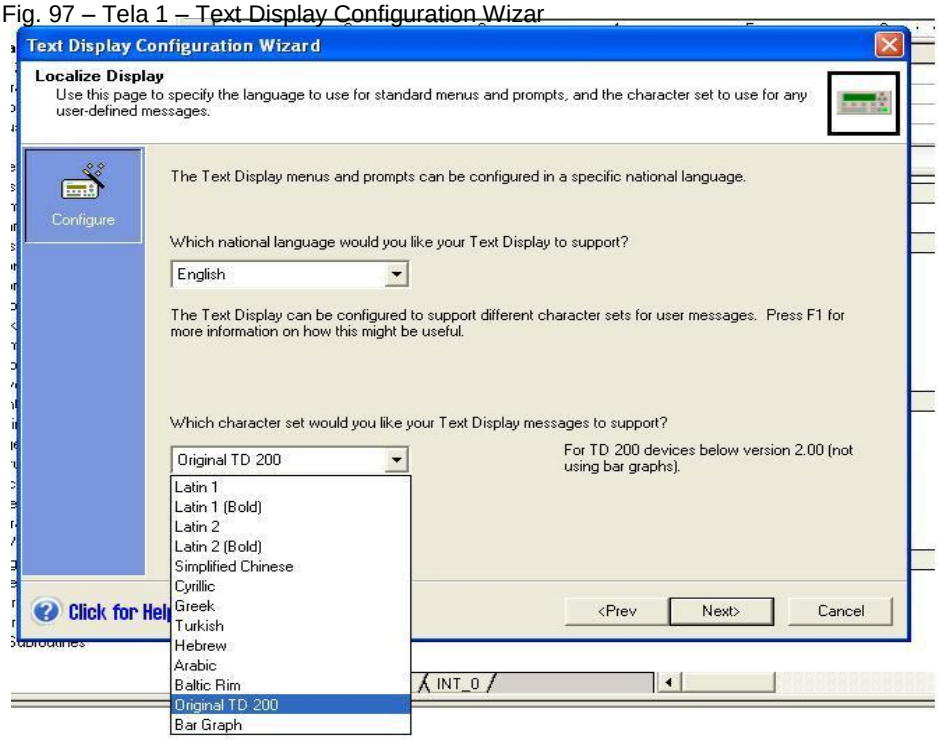

Text Display Wizard
Select TD Model and Version
Use this page to select the model and version of your TD device.
Configure
To view the specific features supported by your TD, please select the model and version of TD you have. To determine the model and version of your TD, you may power-up the unit. The initial screen will display the model and version. The model and version may also be listed on the back of your TD. If there is no version listed on the back of the unit, then select 'Version 2.1 and earlier', below.
Select TD Model and Version
TD 200 Version 2.1 and earlier
TD 200 Version 3.0
TD 200C Version 1.0
TD 100C Version 1.0
Original TD 200
Click for Help and Support
<Prev
Next>
Cancel

Fig. 96 – Tela Text Display Wizard

4. No item “Which national language would you like your Text Display to support?” (qual idioma você gostaria de utilizar em seu display?), selecionar a opção “English”.

No item “Which character set would you like your Text Display messages to support?” (qual tipo de letra você gostaria de usar em suas mensagens?), selecionar a opção “Original TD 200” e Next.

Fig. 97 – Tela 1 – Text Display Configuration Wizar


Text Display Configuration Wizard
Localize Display
Use this page to specify the language to use for standard menus and prompts, and the character set to use for any user-defined messages.
Configure
The Text Display menus and prompts can be configured in a specific national language.
Which national language would you like your Text Display to support?
English
The Text Display can be configured to support different character sets for user messages. Press F1 for more information on how this might be useful.
Which character set would you like your Text Display messages to support?
Original TD 200
For TD 200 devices below version 2.00 (not using bar graphs).
Latin 1
Latin 1 (Bold)
Latin 2
Latin 2 (Bold)
Simplified Chinese
Cyrillic
Greek
Turkish
Hebrew
Arabic
Baltic Rim
Original TD 200
Bar Graph
<Prev
Next>
Cancel
INT_0

4. Nesta etapa vamos escolher se desejamos ou não habilitar os seguintes menus da IHM:

- Hora e Dia (time of day)
- Forçar variáveis (force menu)
- Senha de proteção (password protection). Neste caso, podemos usar uma senha de 4 (quato) dígitos. Após selecionar, clique Next.

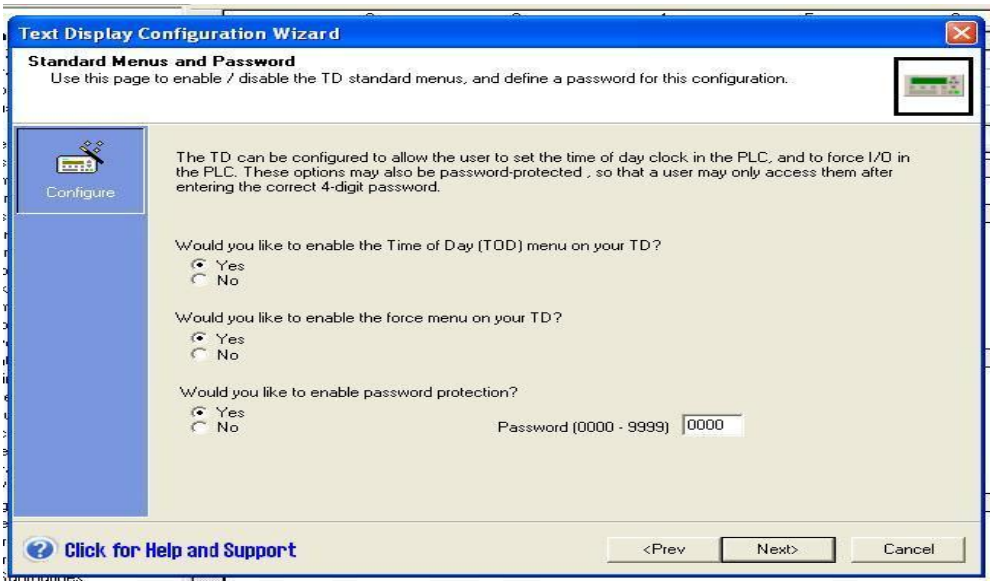
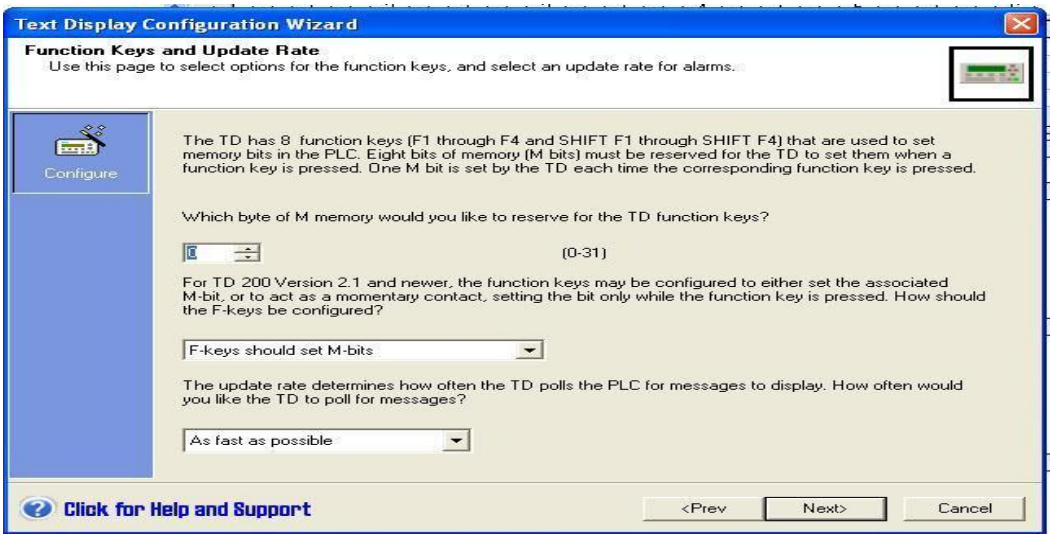
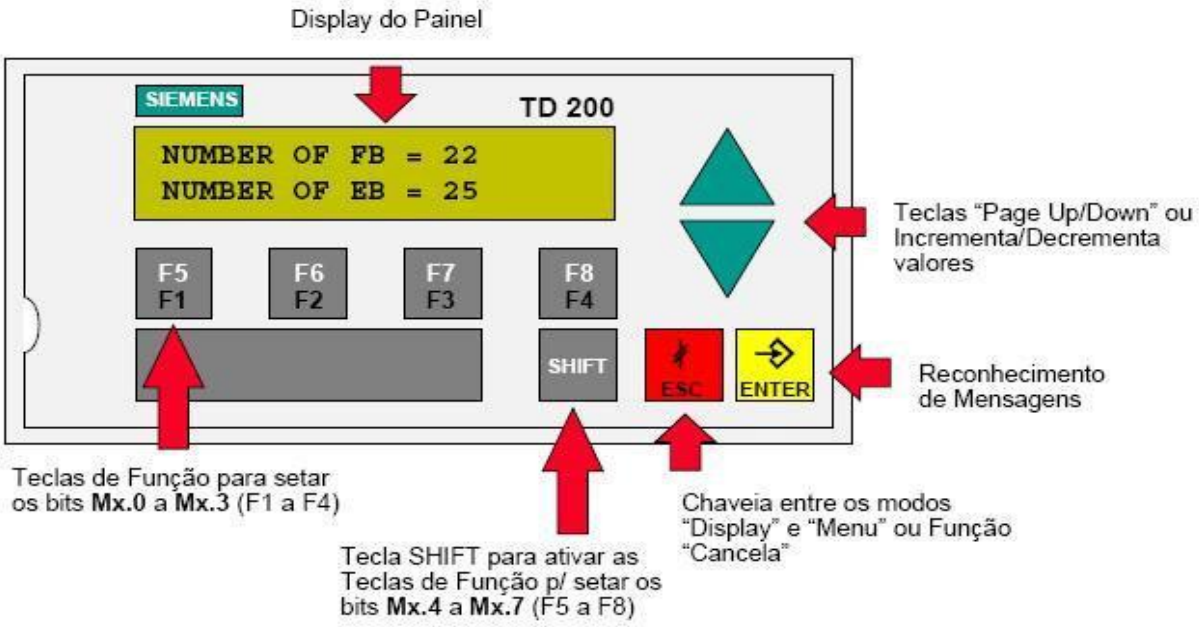

Text Display Configuration Wizard
Standard Menus and Password
Use this page to enable / disable the TD standard menus, and define a password for this configuration.
Configure
The TD can be configured to allow the user to set the time of day clock in the PLC, and to force I/O in the PLC. These options may also be password-protected , so that a user may only access them after entering the correct 4-digit password.
Would you like to enable the Time of Day (TOD) menu on your TD?
Yes
No
Would you like to enable the force menu on your TD?
Yes
No
Would you like to enable password protection?
Yes
No
Password (0000 - 9999) 0000
Click for Help and Support
<Prev
Next>
Cancel

Fig. 98 – Tela 2 - Text Display Configuration Wizard

5. A IHM TD 200 possui 8 teclas de funções (F1 a F4 e SHIFT F1 a SHIFT F4), conforme representado na tela abaixo:


Text Display Configuration Wizard
Function Keys and Update Rate
Use this page to select options for the function keys, and select an update rate for alarms.
Configure
The TD has 8 function keys (F1 through F4 and SHIFT F1 through SHIFT F4) that are used to set memory bits in the PLC. Eight bits of memory (M bits) must be reserved for the TD to set them when a function key is pressed. One M bit is set by the TD each time the corresponding function key is pressed.
Which byte of M memory would you like to reserve for the TD function keys?
0
(0-31)
For TD 200 Version 2.1 and newer, the function keys may be configured to either set the associated M-bit, or to act as a momentary contact, setting the bit only while the function key is pressed. How should the F-keys be configured?
F-keys should set M-bits
The update rate determines how often the TD polls the PLC for messages to display. How often would you like the TD to poll for messages?
As fast as possible
Click for Help and Support
<Prev
Next>
Cancel

ig. 99 - Tela 3 – Text Display

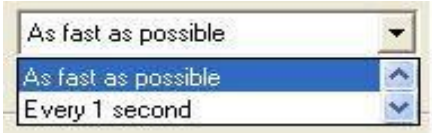


Configuration Wizard - Fig. 100 – Display do TD 200

As teclas F1 a F8 estão associadas à área de memória M. Temos a opção de escolher um byte desta área no intervalo de 0 a 31. Cada bit do byte escolhido estará associado a uma tecla de função da IHM, por exemplo: se escolhermos o byte M0, a tecla F1 estará associada ao bit M0.0; a tecla F2 estará associada ao bit M0.1 e assim por diante.

6. Após ter escolhido o byte da memória M que estará associado às teclas da TD 200, precisamos agora escolher se essas teclas vão "setar" os bits que estão associados a elas (F-keys should set M-bits) ou se vão ser contatos momentâneos, nível lógico apenas se a tecla estiver sendo pressionada (F-keys as momentary contacts).

Fig. 101 – Opção F-Keys...



Em seguida, next.

Fig. 102 – Opção as fast as...

8. Nesta etapa escolheremos a quantidade de caracteres para as mensagens (messages sizes) que mostraremos na tela do TD 200 e a quantidade de mensagens (how many messages).
Podemos configurar até 80 mensagens para o TD 200. Com 20 caracteres na mensagem, podemos mostrar duas mensagens por vez no display, e com 40 caracteres apenas uma por vez. Em seguida, next.

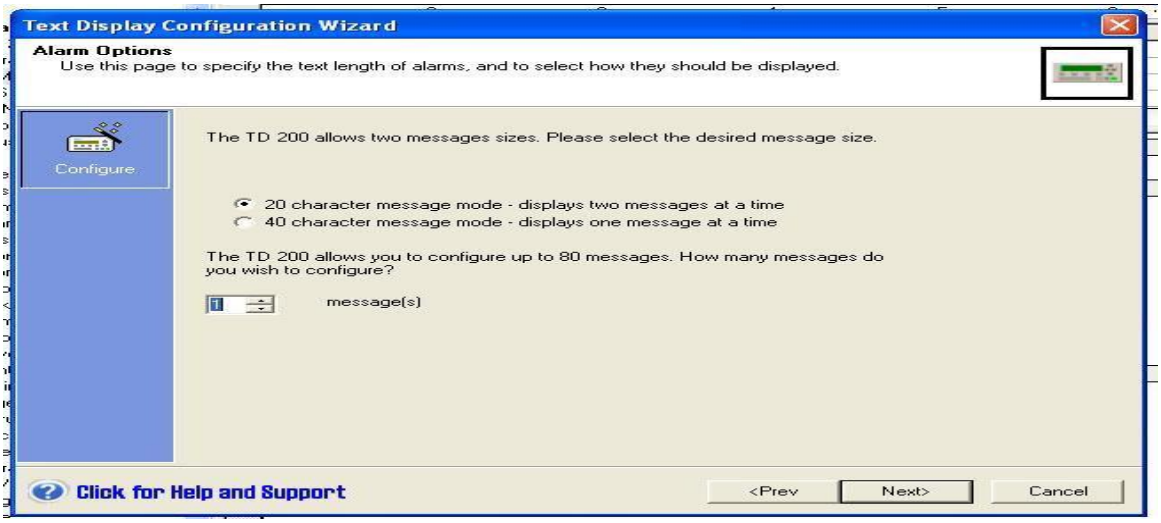

Text Display Configuration Wizard
Alarm Options
Use this page to specify the text length of alarms, and to select how they should be displayed.
Configure
The TD 200 allows two messages sizes. Please select the desired message size.
20 character message mode - displays two messages at a time
40 character message mode - displays one message at a time
The TD 200 allows you to configure up to 80 messages. How many messages do you wish to configure?
1
message(s)
Click for Help and Support
<Prev
Next>
Cancel

Fig. 103 – Tela 4 - Text Display Configuration Wizard

9. Agora vamos definir a área de memória que será utilizada para armazenar os dados do parâmetro block.

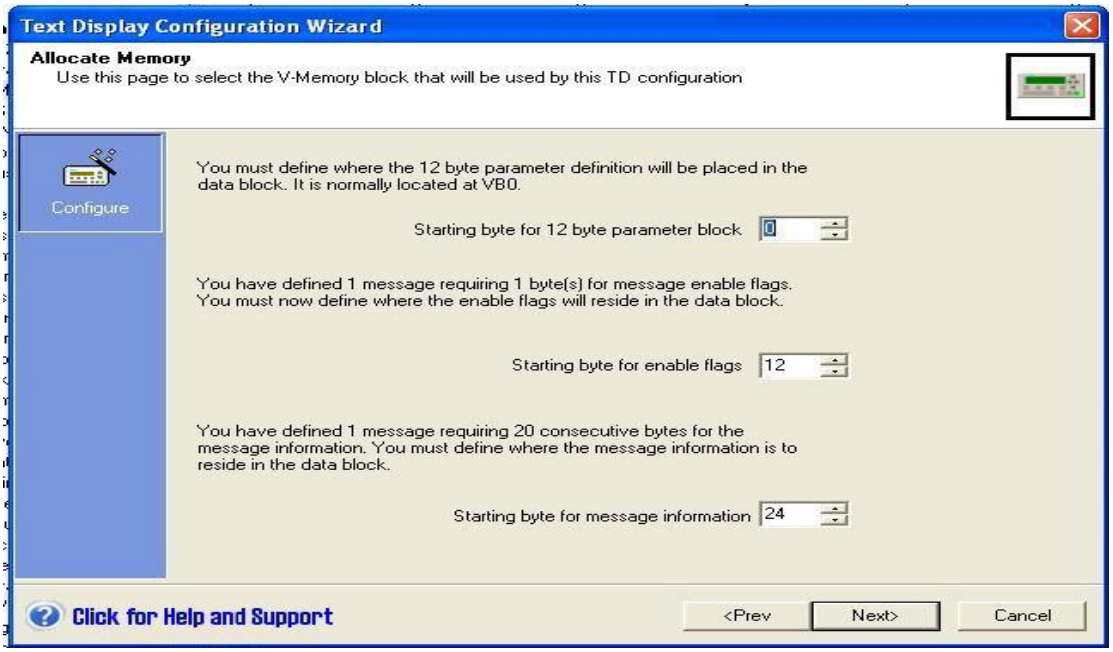

Text Display Configuration Wizard
Allocate Memory
Use this page to select the V-Memory block that will be used by this TD configuration
Configure
You must define where the 12 byte parameter definition will be placed in the data block. It is normally located at VB0.
Starting byte for 12 byte parameter block
0
You have defined 1 message requiring 1 byte(s) for message enable flags. You must now define where the enable flags will reside in the data block.
Starting byte for enable flags
12
You have defined 1 message requiring 20 consecutive bytes for the message information. You must define where the message information is to reside in the data block.
Starting byte for message information
24
Click for Help and Support
<Prev
Next>
Cancel

Fig. 104 – Tela 5 - Text Display Configuration Wizard

Precisamos definir o byte inicial para alocação dos dados do parâmetro block. Para tanto são necessários 12 bytes da área de memória V. Se definimos o byte 0 como inicial, o próximo byte disponível para utilização será o 12.

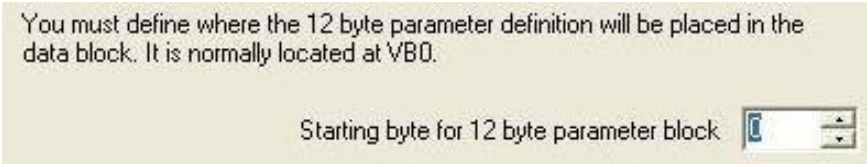

You must define where the 12 byte parameter definition will be placed in the data block. It is normally located at VB0.
Starting byte for 12 byte parameter block
0

Fig. 105 – Configuração do TD 200 (1)

Em seguida, precisamos definir o byte ou bytes, dependendo da quantidade de mensagens, onde estarão os bits que serão responsáveis por habilitar as mensagens (enable flag).

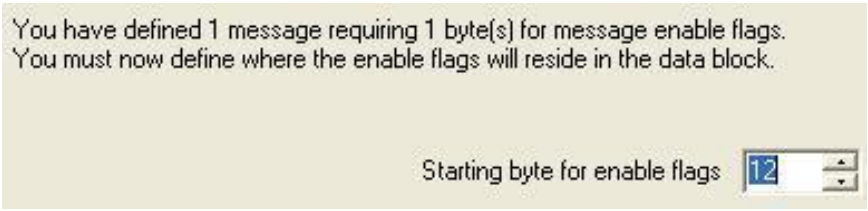

You have defined 1 message requiring 1 byte(s) for message enable flags.
You must now define where the enable flags will reside in the data block.
Starting byte for enable flags
12

Fig. 106 – Configuração do TD 200 (2)

10. Nosso próximo passo será definir a área de memória que será utilizada para armazenar os dados (caracteres) que estarão contidos na nossa mensagem. Como anteriormente definimos apenas uma mensagem de 20 caracteres, são necessários 20 bytes consecutivos para armazenar estes caracteres. Neste caso definimos como byte inicial o 14, então, o próximo disponível será o byte 34. A seguir, next.

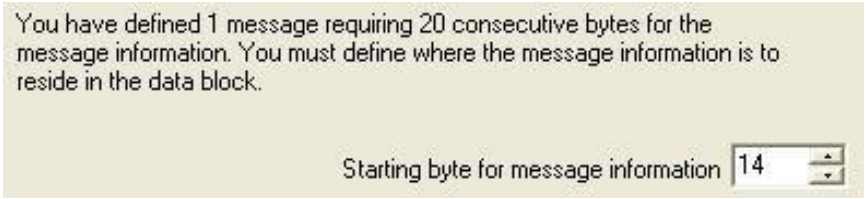

You have defined 1 message requiring 20 consecutive bytes for the message information. You must define where the message information is to reside in the data block.
Starting byte for message information
14

Fig. 107 – Configuração do TD 200 (3)

11. Agora precisamos apenas escrever a nossa mensagem.
O bit responsável por chamar a mensagem será o V12.7, conforme informado no campo message enabled bit (bit que habilita a mensagem).

Fig. 108 – Configuração do TD 200 (4)

Vamos testar? Execute a rotina, a seguir, no CLP e analise resultado

Desenvolver um programa para uma chave de partida direta e mostrar na tela da TD 200 as mensagens:

- Motor desligado;
- Motor ligado;
- Sobrecarga no Motor.

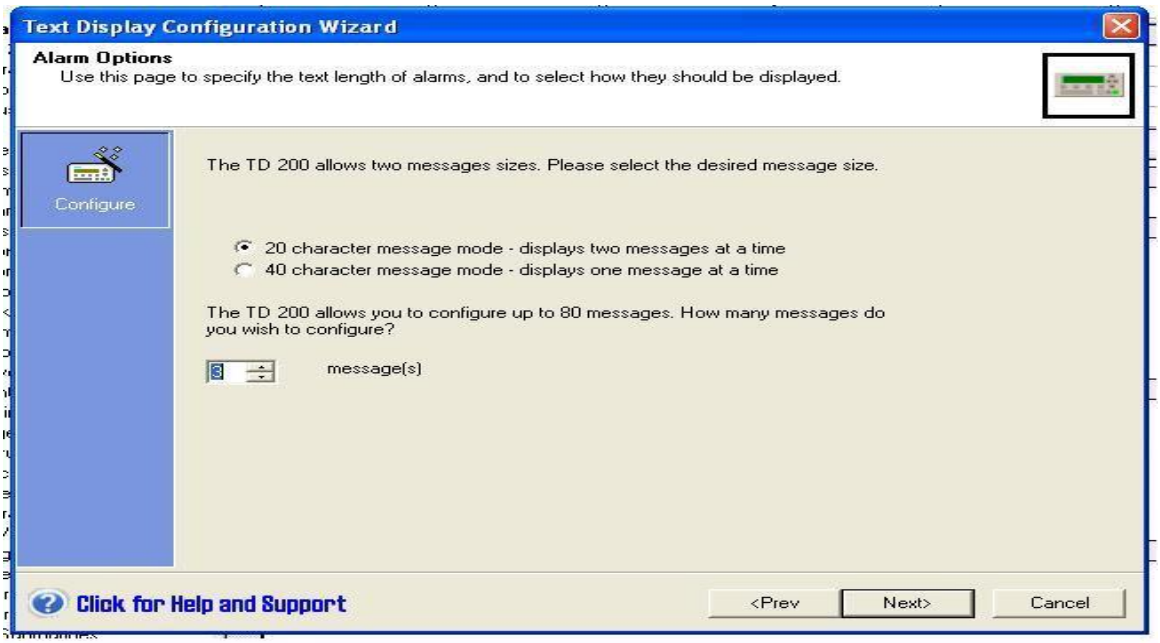

Text Display Configuration Wizard
Alarm Options
Use this page to specify the text length of alarms, and to select how they should be displayed.
Configure
The TD 200 allows two messages sizes. Please select the desired message size.
20 character message mode - displays two messages at a time
40 character message mode - displays one message at a time
The TD 200 allows you to configure up to 80 messages. How many messages do you wish to configure?
3
message(s)
Click for Help and Support
<Prev
Next>
Cancel

Fig. 109 – Configuração do TD 200 (5)

Solução:

- Seguiremos o roteiro anterior até a etapa nº 08;
- Na etapa nº10 definiremos o número de mensagens = 3, seguindo a orientação do exercício.
- Na etapa nº10 faremos como segue.

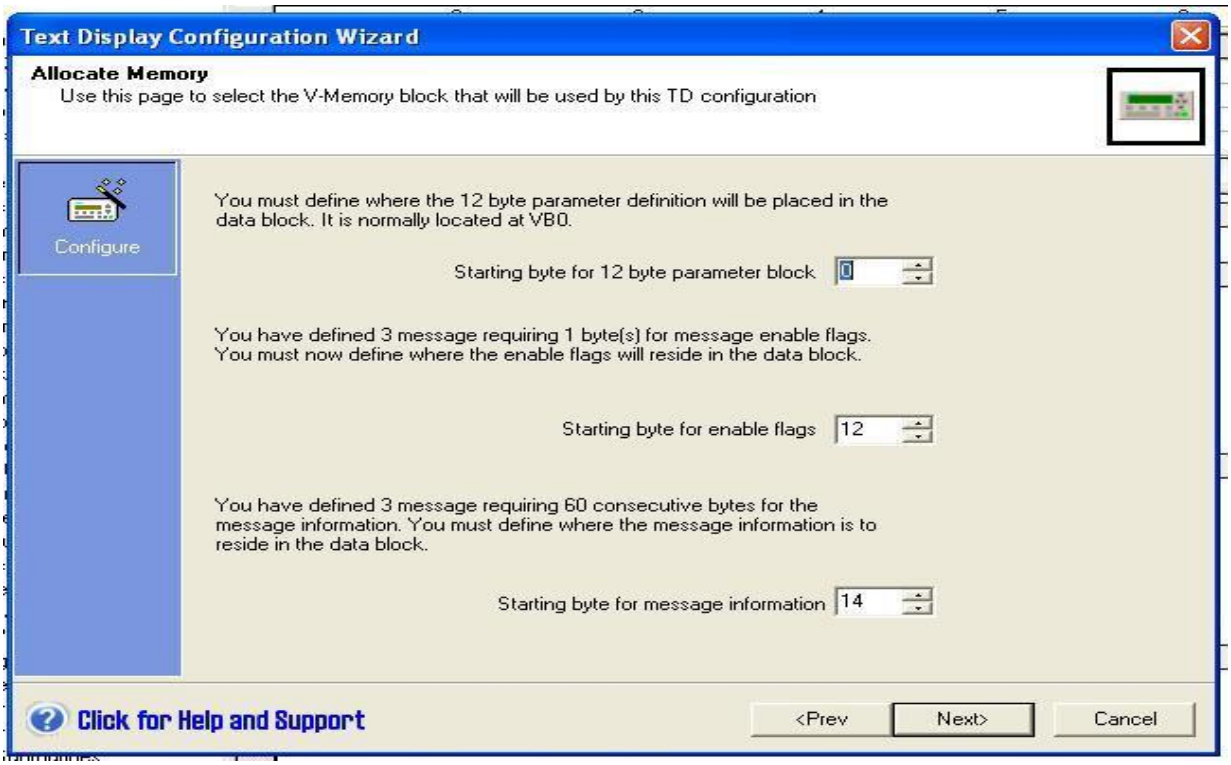

Text Display Configuration Wizard
Allocate Memory
Use this page to select the V-Memory block that will be used by this TD configuration
Configure
You must define where the 12 byte parameter definition will be placed in the data block. It is normally located at VB0.
Starting byte for 12 byte parameter block
0
You have defined 3 message requiring 1 byte(s) for message enable flags. You must now define where the enable flags will reside in the data block.
Starting byte for enable flags
12
You have defined 3 message requiring 60 consecutive bytes for the message information. You must define where the message information is to reside in the data block.
Starting byte for message information
14
Click for Help and Support
<Prev
Next>
Cancel

Fig. 110 – Configuração do TD 200 (solução do exercício) – (1A)

- A etapa nº11 continua igual.
- Chegando à etapa nº12, colocaremos as mensagens sugeridas no problema. Observe que a tela de diálogo pede mensagem 1 de 3, depois 2 de 3 e, por último, 3 de 3.

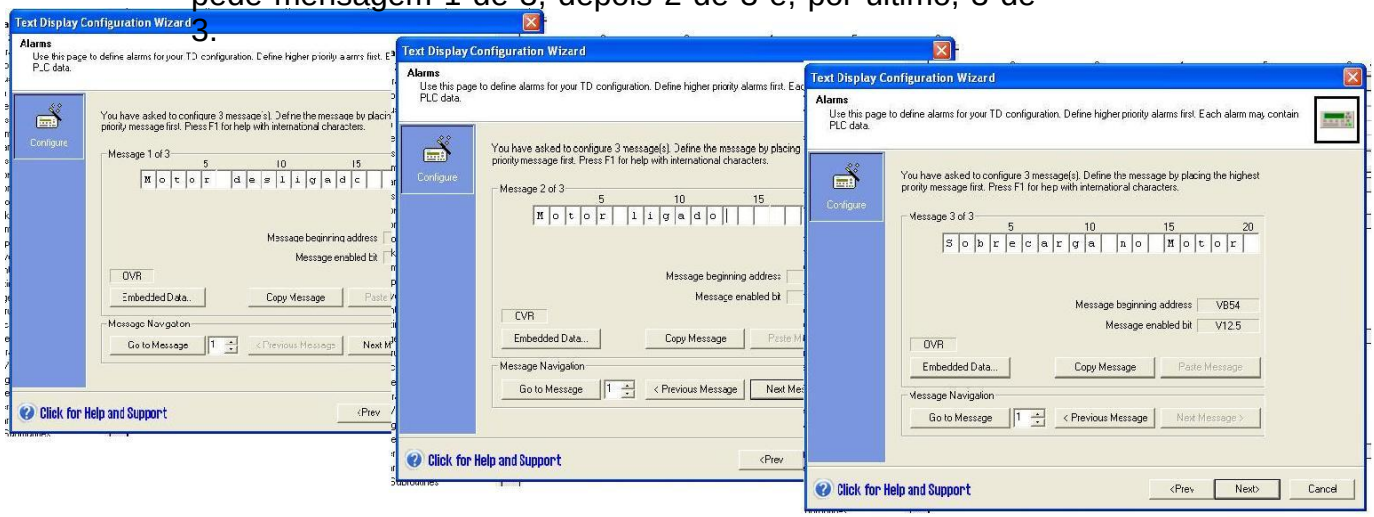


Fig. 111 - Configuração do TD 200 (solução do exercício)

- Entre as mensagens sempre usar NEXT

Programa:

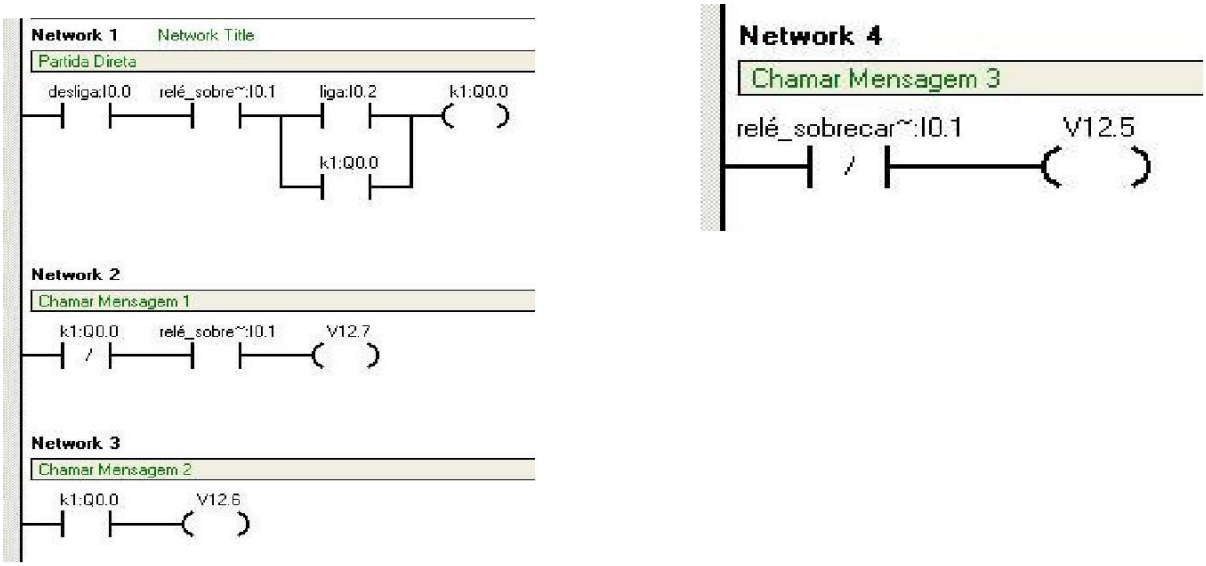


Fig. 112 – Rotina de programação (solução do exercício) – (3A)

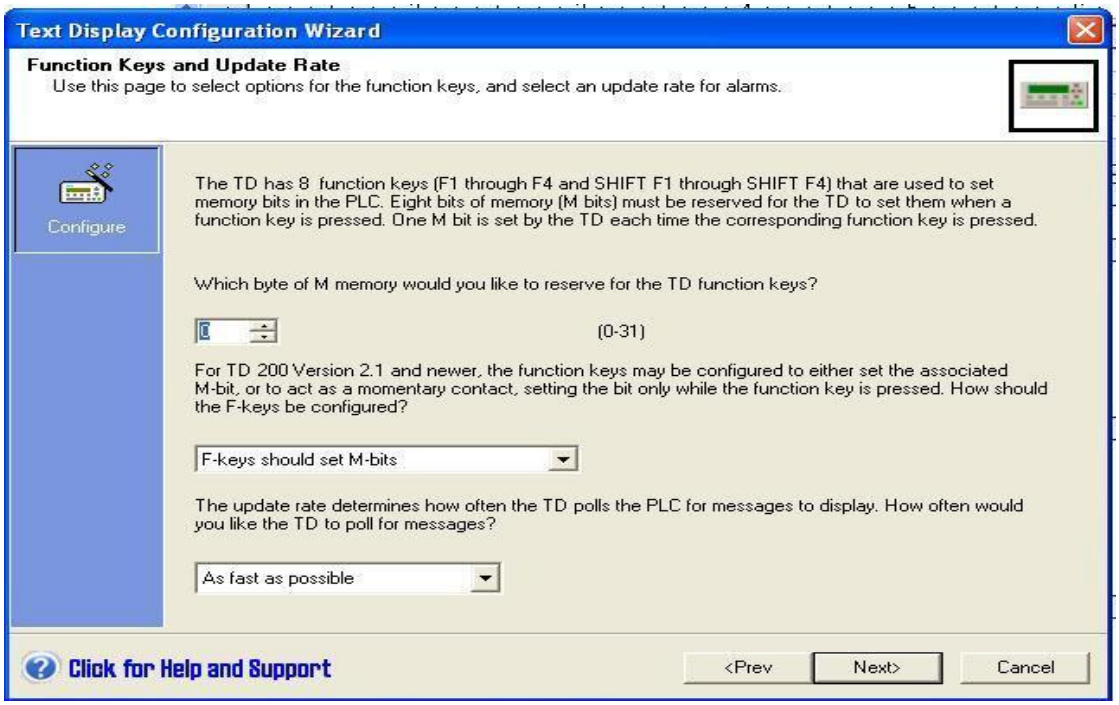

Text Display Configuration Wizard
Function Keys and Update Rate
Use this page to select options for the function keys, and select an update rate for alarms.
Configure
The TD has 8 function keys (F1 through F4 and SHIFT F1 through SHIFT F4) that are used to set memory bits in the PLC. Eight bits of memory (M bits) must be reserved for the TD to set them when a function key is pressed. One M bit is set by the TD each time the corresponding function key is pressed.
Which byte of M memory would you like to reserve for the TD function keys?
0
(0-31)
For TD 200 Version 2.1 and newer, the function keys may be configured to either set the associated M-bit, or to act as a momentary contact, setting the bit only while the function key is pressed. How should the F-keys be configured?
F-keys should set M-bits
The update rate determines how often the TD polls the PLC for messages to display. How often would you like the TD to poll for messages?
As fast as possible
Click for Help and Support
<Prev
Next>
Cancel

Fig. 113 - Configuração do TD 200 (solução do exercício) – (1B)

Vamos testar? Execute a rotina, a seguir, no CLP e analise o resultado

Desenvolver um programa para uma chave de partida direta, utilizando a tecla F1 do TD 200 para ligar, e a tecla F2 para desligar o motor e mostrar na tela do TD 200 as mensagens:

- Motor desligado;
- Motor ligado;
- Sobrecarga no Motor.

O processo é similar.
Definimos na tela, a seguir, a utilização da memória M0 associada às teclas de atalho de F1 a F8, de tal forma que as teclas que queremos utilizar, F1 e F2, estarão associadas aos bits M0.0 e M0.1.

Agora definimos que serão 3 mensagens.

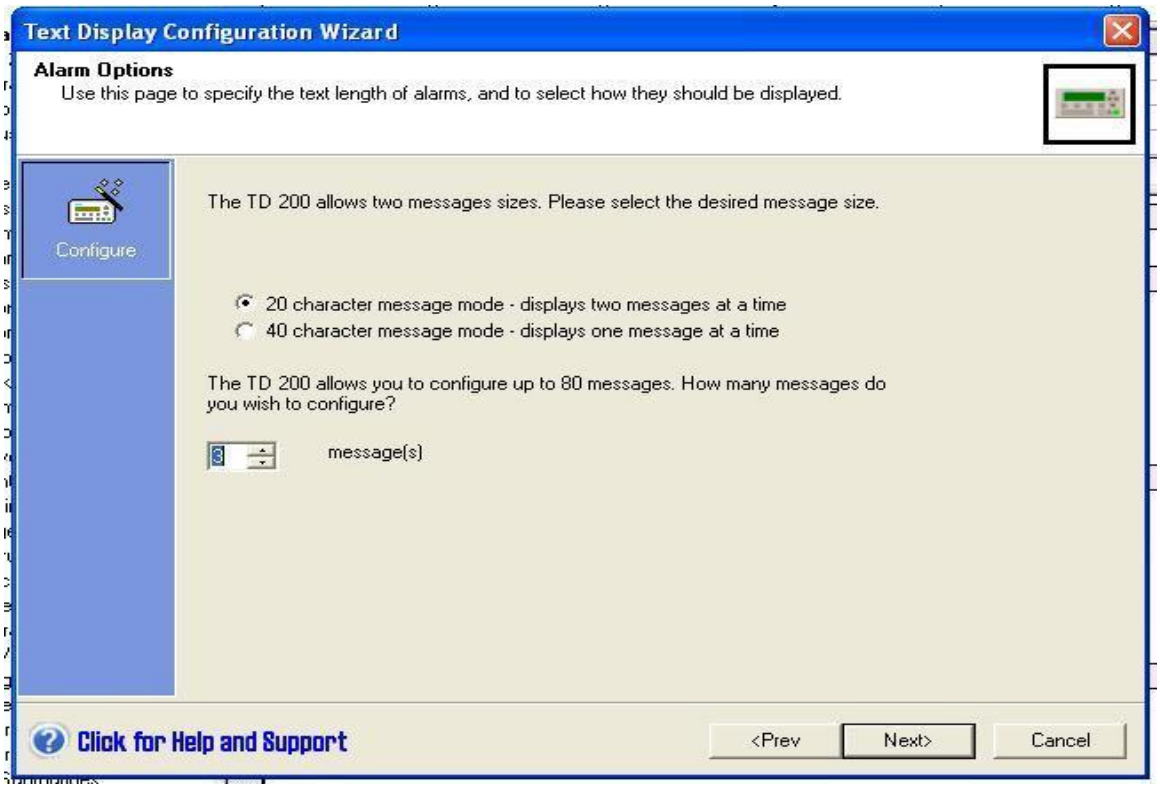

Text Display Configuration Wizard
Alarm Options
Use this page to specify the text length of alarms, and to select how they should be displayed.
Configure
The TD 200 allows two messages sizes. Please select the desired message size.
20 character message mode - displays two messages at a time
40 character message mode - displays one message at a time
The TD 200 allows you to configure up to 80 messages. How many messages do you wish to configure?
3
message(s)
Click for Help and Support
<Prev
Next>
Cancel

Fig. 114 - Configuração do TD 200 (solução do exercício) – (2B)

Depois os bytes que serão utilizados.

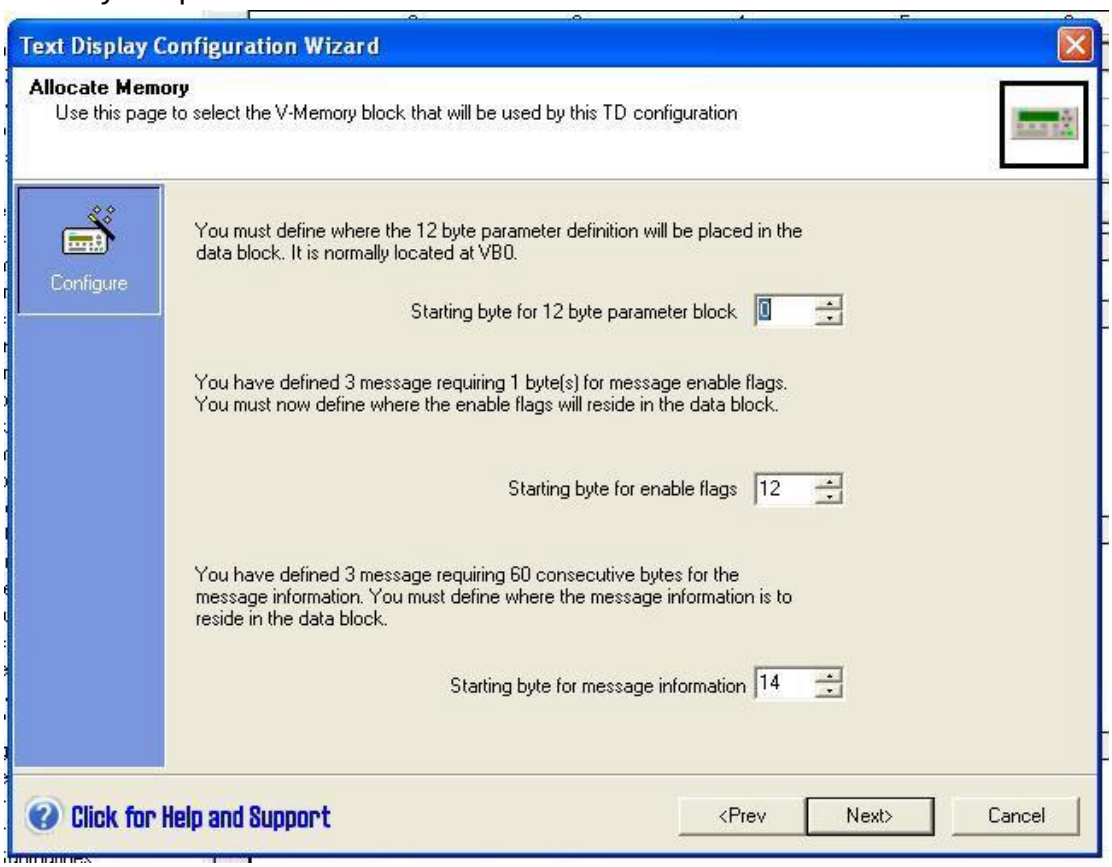

Text Display Configuration Wizard
Allocate Memory
Use this page to select the V-Memory block that will be used by this TD configuration
Configure
You must define where the 12 byte parameter definition will be placed in the data block. It is normally located at VB0.
Starting byte for 12 byte parameter block
0
You have defined 3 message requiring 1 byte(s) for message enable flags. You must now define where the enable flags will reside in the data block.
Starting byte for enable flags
12
You have defined 3 message requiring 60 consecutive bytes for the message information. You must define where the message information is to reside in the data block.
Starting byte for message information
14
Click for Help and Support
<Prev
Next>
Cancel

Fig. 115 - Configuração do TD 200 (solução do exercício) – (3B)

Depois digitamos as mensagens.

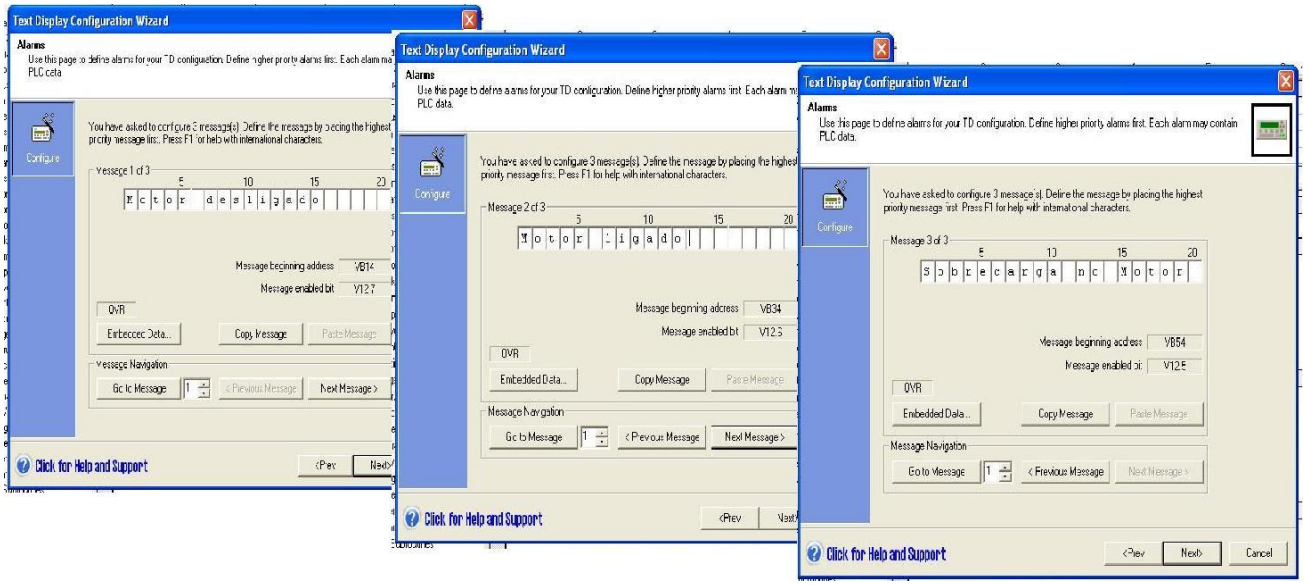

Text Display Configuration Wizard
Alarms
Use this page to define alarms for your TD configuration. Define higher priority alarms first. Each alarm may contain PLC data.
Configure
Click for Help and Support
Cancel

Fig. 116 - Configuração do TD 200 (solução do exercício) – (4B)

Programa:

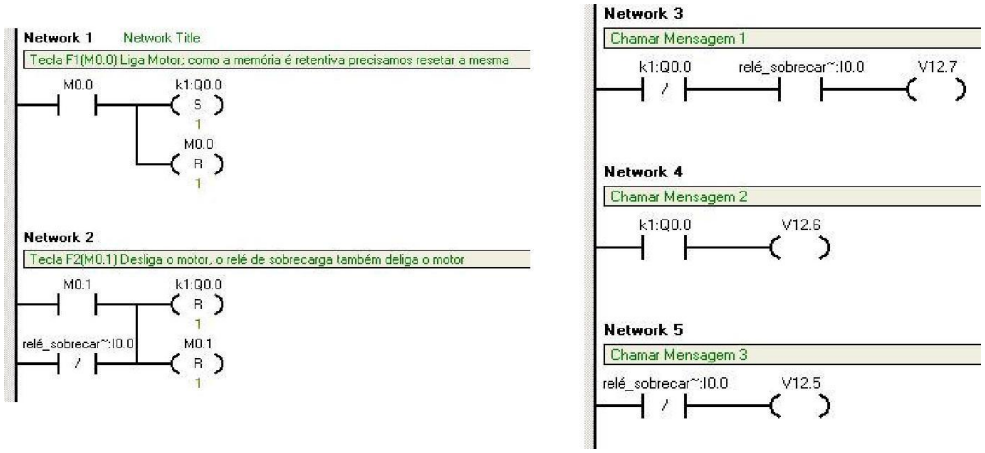


Fig. 117 – Rotina de programação (solução do exercício) – (5B)

Vamos testar? Execute a rotina, a seguir, no CLP e analise o resultado

Desenvolver um programa de forma que possa ser visto na tela do TD 200 o valor de contagem do contador crescente C0.

Vamos utilizar a entrada I0.0 para enviar os pulsos de contagem para o contador, a entrada I0.1 para resetar a contagem, a tecla F1 para mostrar na tela do TD 200 o valor da contagem e a tecla F2 para retirar a mensagem da tela.

Primeiro definimos a memória M0 associada às teclas de atalho de F1 a F8.

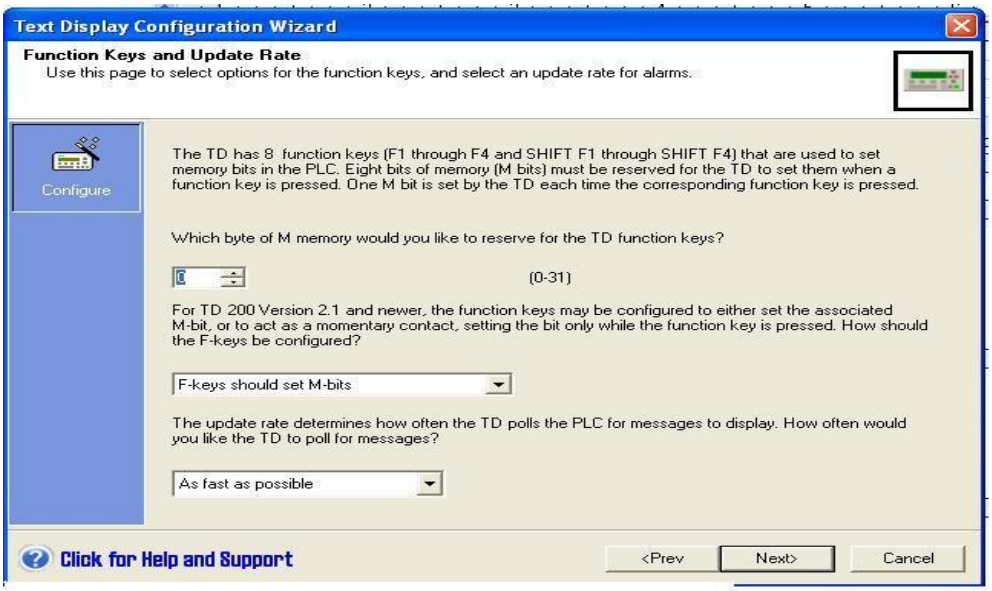

Text Display Configuration Wizard
Function Keys and Update Rate
Use this page to select options for the function keys, and select an update rate for alarms.
Configure
The TD has 8 function keys (F1 through F4 and SHIFT F1 through SHIFT F4) that are used to set memory bits in the PLC. Eight bits of memory (M bits) must be reserved for the TD to set them when a function key is pressed. One M bit is set by the TD each time the corresponding function key is pressed.
Which byte of M memory would you like to reserve for the TD function keys?
0
(0-31)
For TD 200 Version 2.1 and newer, the function keys may be configured to either set the associated M-bit, or to act as a momentary contact, setting the bit only while the function key is pressed. How should the F-keys be configured?
F-keys should set M-bits
The update rate determines how often the TD polls the PLC for messages to display. How often would you like the TD to poll for messages?
As fast as possible
Click for Help and Support
<Prev
Next>
Cancel

Fig. 118 – Configuração do TD 200 (solução do exercício)

Agora definimos que será uma mensagem.

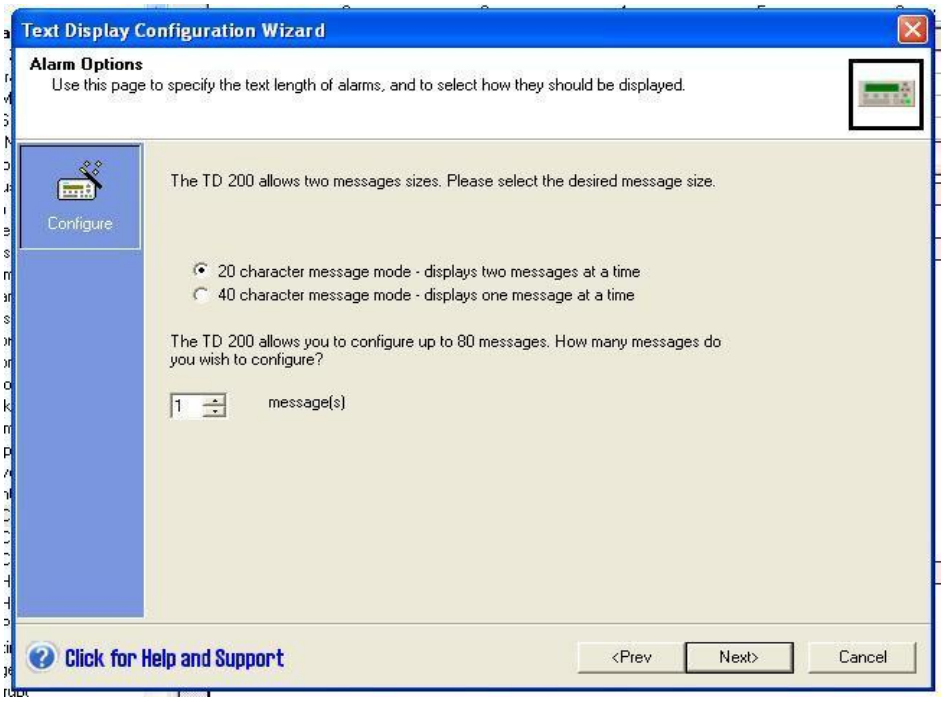
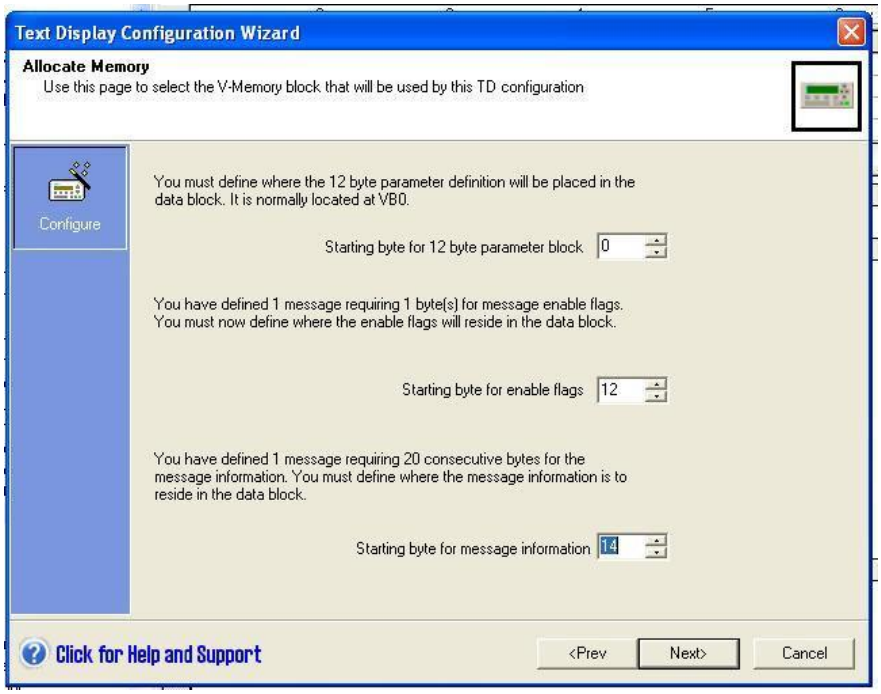

Text Display Configuration Wizard
Alarm Options
Use this page to specify the text length of alarms, and to select how they should be displayed.
Configure
The TD 200 allows two messages sizes. Please select the desired message size.
20 character message mode - displays two messages at a time
40 character message mode - displays one message at a time
The TD 200 allows you to configure up to 80 messages. How many messages do you wish to configure?
1
message(s)
Click for Help and Support
<Prev
Next>
Cancel

Fig. 119 – Configuração do TD 200 (solução do exercício) – (2C)

Fig. 120 – Configuração do TD 200 (solução do exercício) – (3C)


Text Display Configuration Wizard
Allocate Memory
Use this page to select the V-Memory block that will be used by this TD configuration
Configure
You must define where the 12 byte parameter definition will be placed in the data block. It is normally located at VB0.
Starting byte for 12 byte parameter block
0
You have defined 1 message requiring 1 byte(s) for message enable flags. You must now define where the enable flags will reside in the data block.
Starting byte for enable flags
12
You have defined 1 message requiring 20 consecutive bytes for the message information. You must define where the message information is to reside in the data block.
Starting byte for message information
14
Click for Help and Support
<Prev
Next>
Cancel

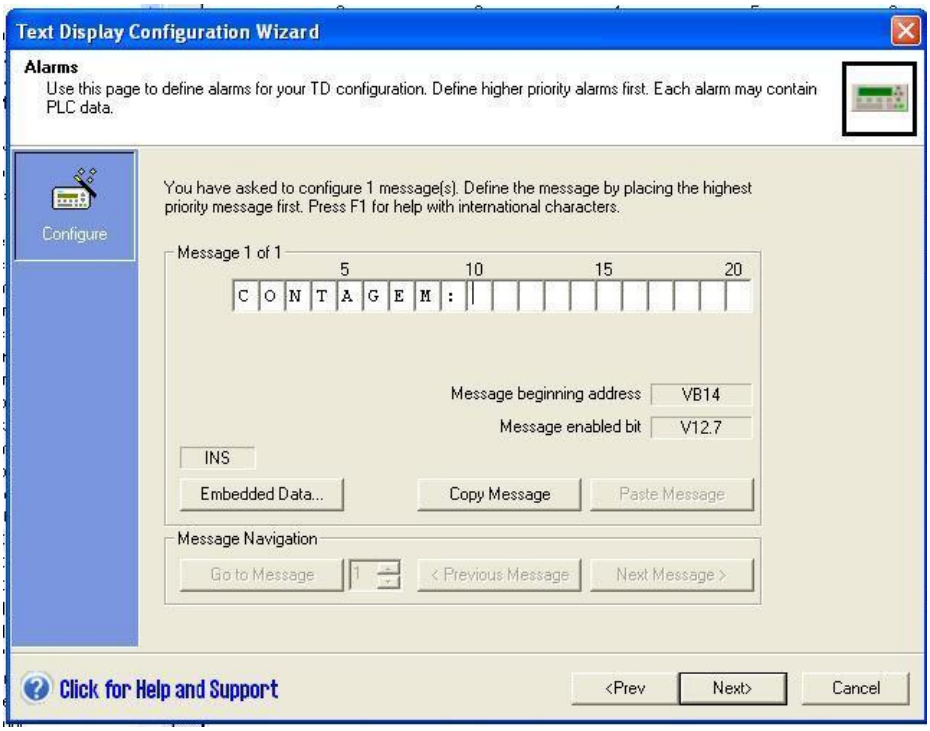

Text Display Configuration Wizard
Alarms
Use this page to define alarms for your TD configuration. Define higher priority alarms first. Each alarm may contain PLC data.
Configure
You have asked to configure 1 message(s). Define the message by placing the highest priority message first. Press F1 for help with international characters.
Message 1 of 1
5
10
15
20
C O N T A G E M :
Message beginning address
VB14
Message enabled bit
V12.7
INS
Embedded Data...
Copy Message
Paste Message
Message Navigation
Go to Message
1
< Previous Message
Next Message >
Click for Help and Support
<Prev
Next>
Cancel

Fig. 121 – Configuração do TD 200 (solução do exercício) – (4C)

Após escrever a mensagem CONTAGEM, precisamos que, no próximo campo, seja mostrado o valor da contagem do contador. Para que isto seja possível devemos utilizar a opção **Embedded Data** (dados embutidos), conforme indicado na figura anterior. Após selecionar esta opção surgirá a janela a seguir:

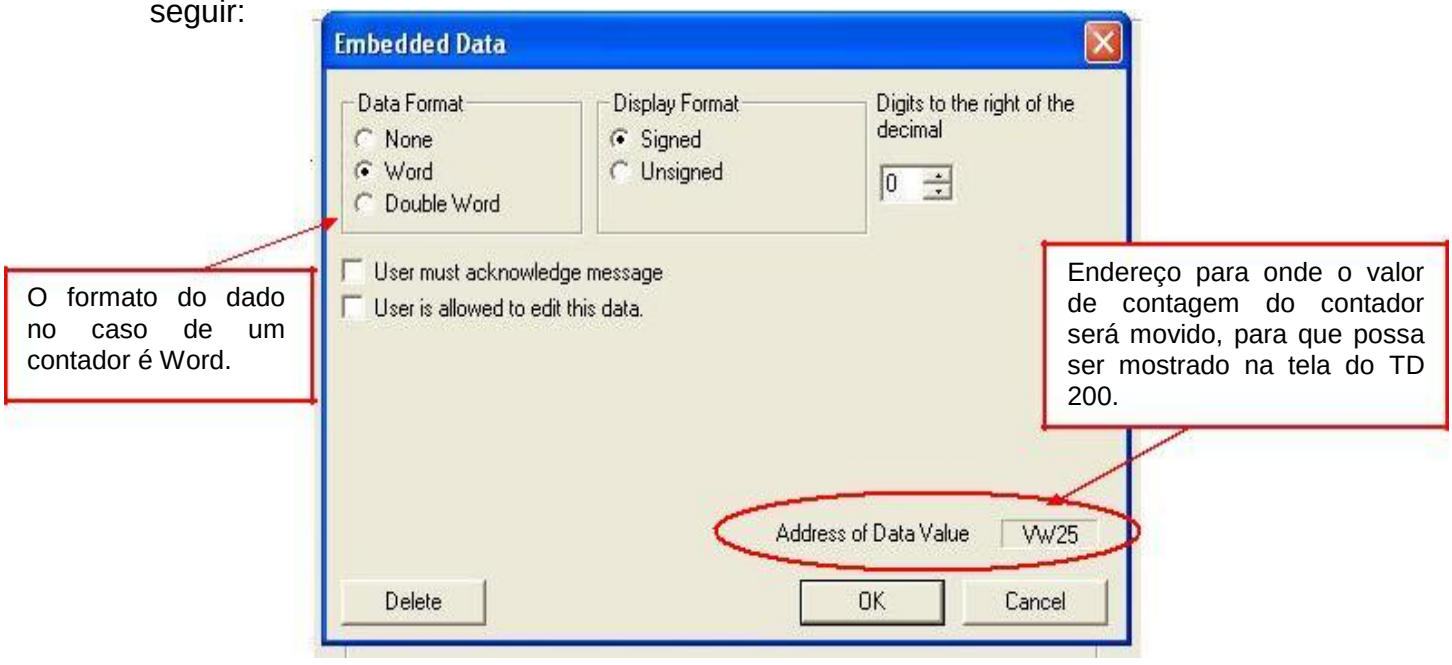


Fig. 122 – Configuração do TD 200 (solução do exercício) – (5C)

Após definir o formato do dado e anotar o endereço para onde o mesmo deve ser movido clicar em OK. Surgirá a janela a seguir:

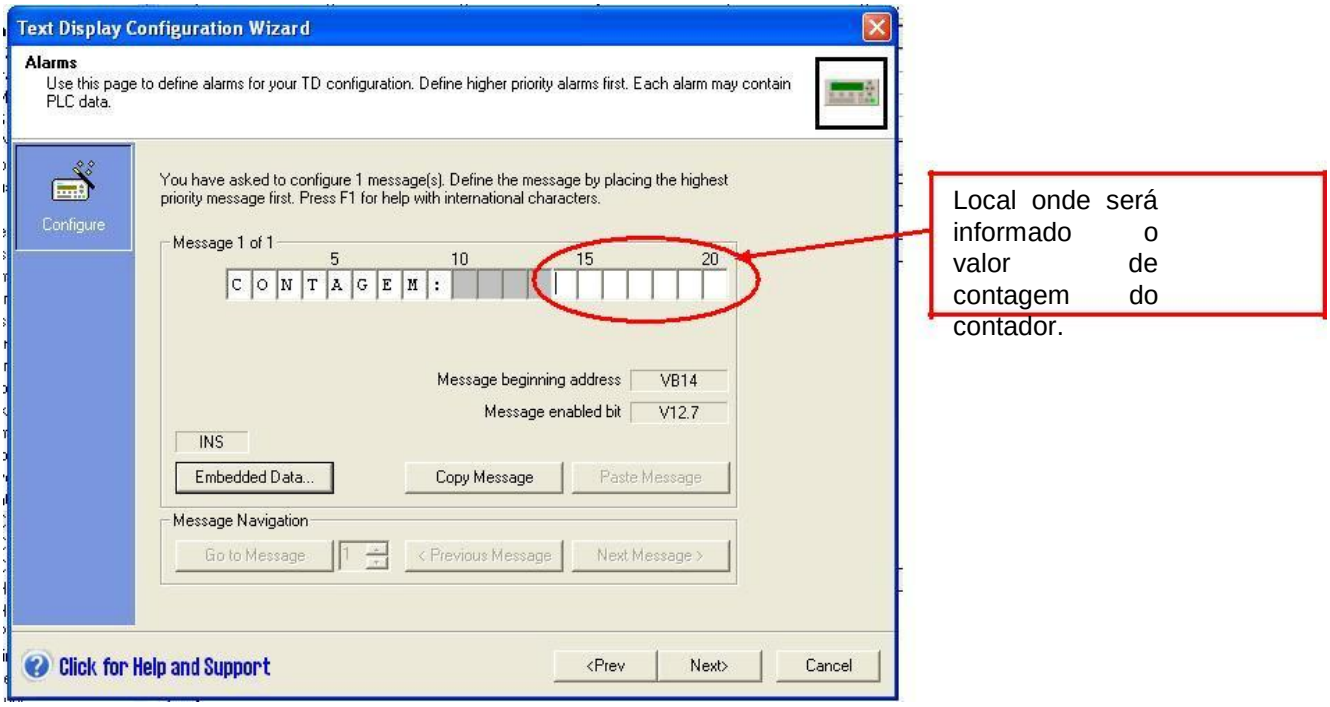
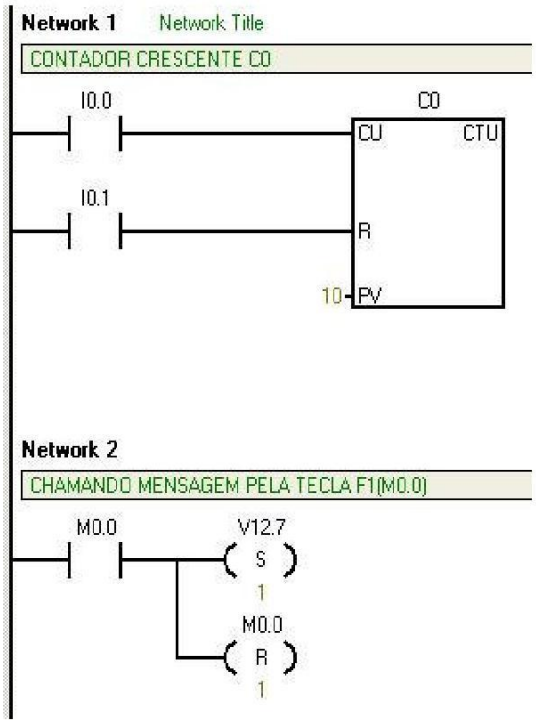

Text Display Configuration Wizard
Alarms
Use this page to define alarms for your TD configuration. Define higher priority alarms first. Each alarm may contain PLC data.
Configure
You have asked to configure 1 message(s). Define the message by placing the highest priority message first. Press F1 for help with international characters.
Message 1 of 1
5
10
15
20
C O N T A G E M :
Message beginning address
VB14
Message enabled bit
V12.7
INS
Embedded Data...
Copy Message
Paste Message
Message Navigation
Go to Message
1
< Previous Message
Next Message >
Click for Help and Support
<Prev
Next>
Cancel
Local onde será informado o valor de contagem do contador.

Fig. 123 – Configuração do TD 200 (solução do exercício) – (6C)

Programa:



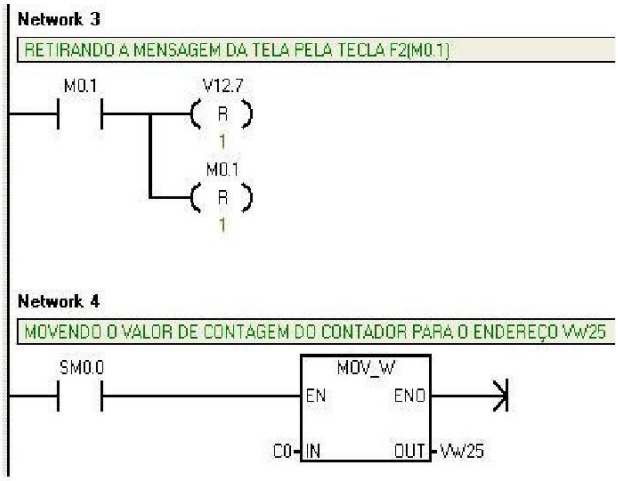


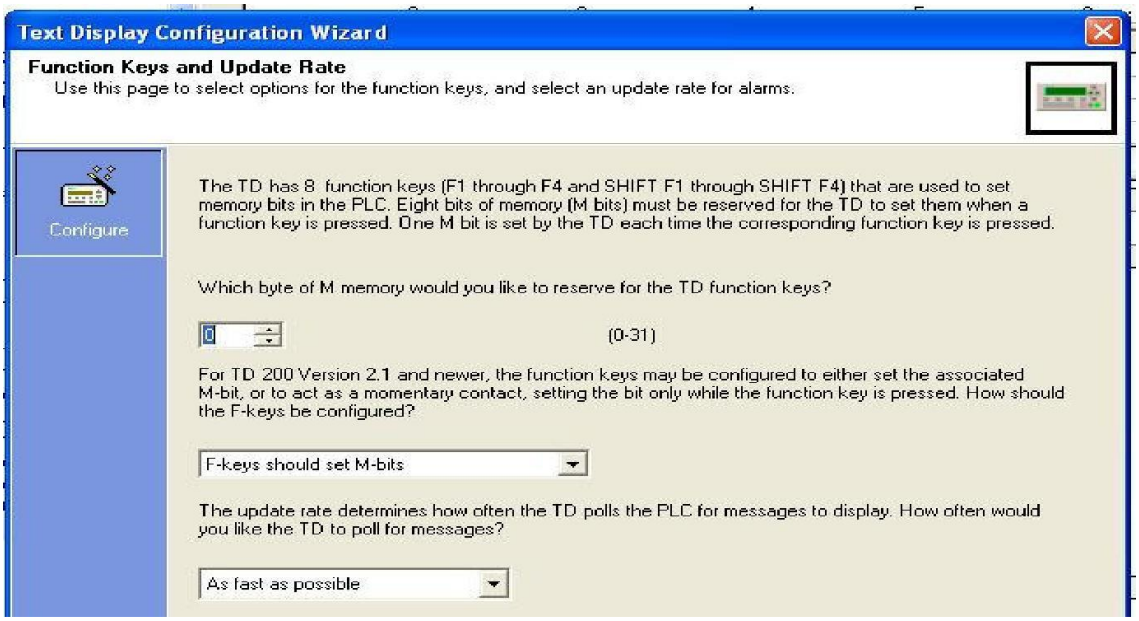
Fig. 124 – Rotina de programação (solução do exercício) – (7C)

Exercício 4

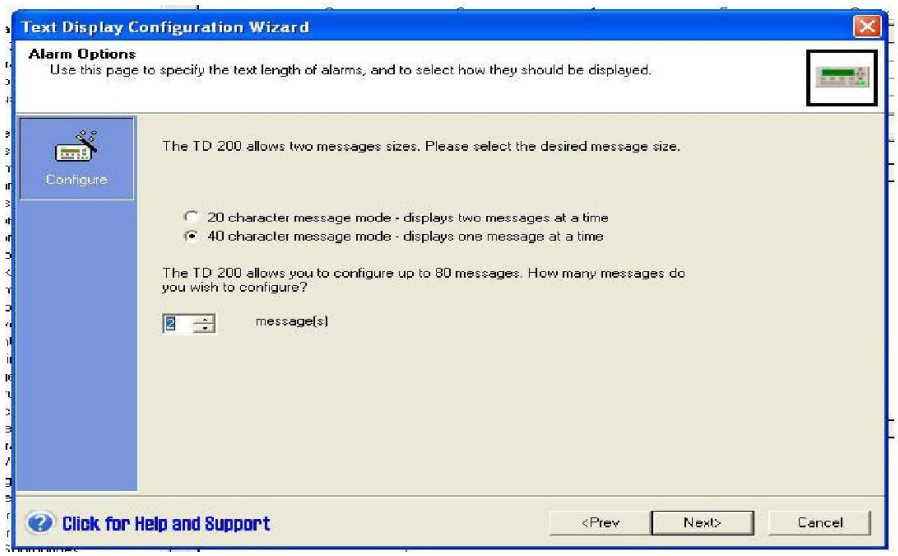
Desenvolver um programa onde o operador possa inserir no contador, via TD 200, a quantidade de peças que serão produzidas, e possa ver na tela da TD 200 o valor atual da produção.

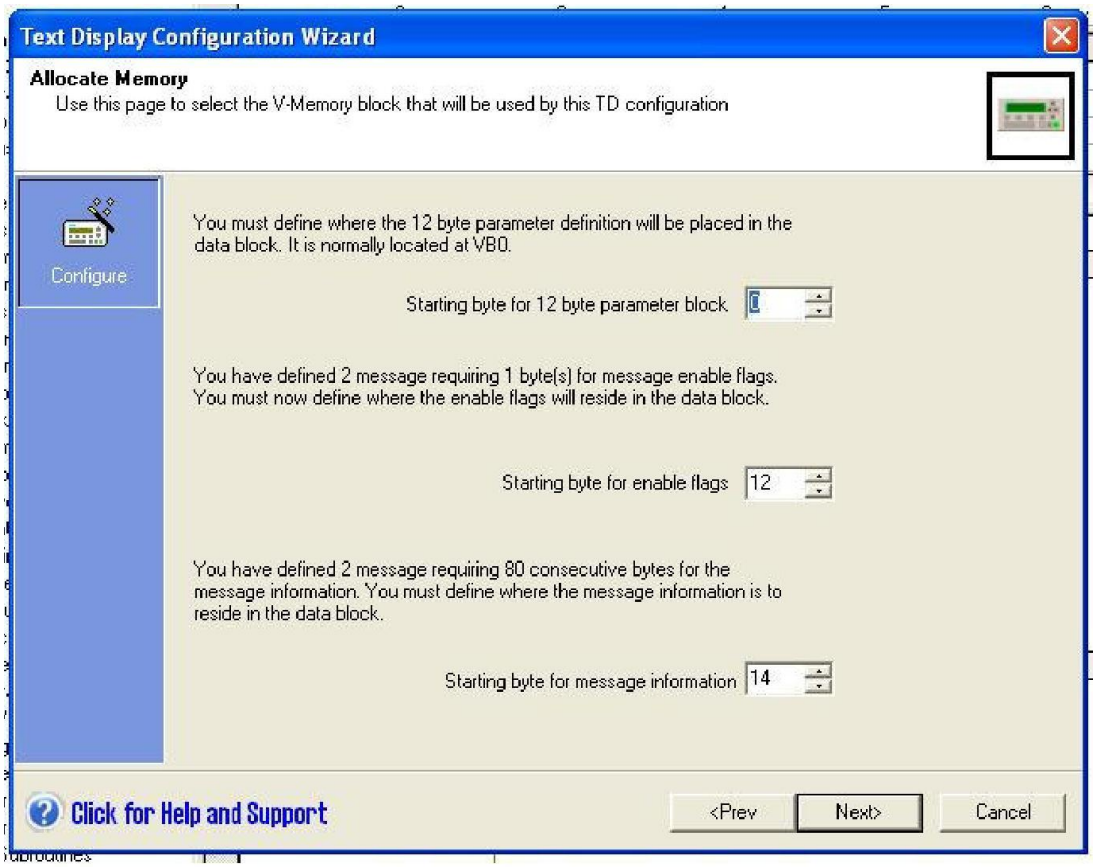
Vamos utilizar a entrada I0.0 para enviar os pulsos de contagem da produção para o contador, a entrada I0.1 para resetar a contagem, a tecla F1 para inserir a quantidade de peças que serão produzidas, a tecla F2 para mostrar na tela da TD 200 o valor da contagem e a tecla F3 para retirar a mensagem da tela. Neste caso, precisaremos definir duas telas de mensagens.

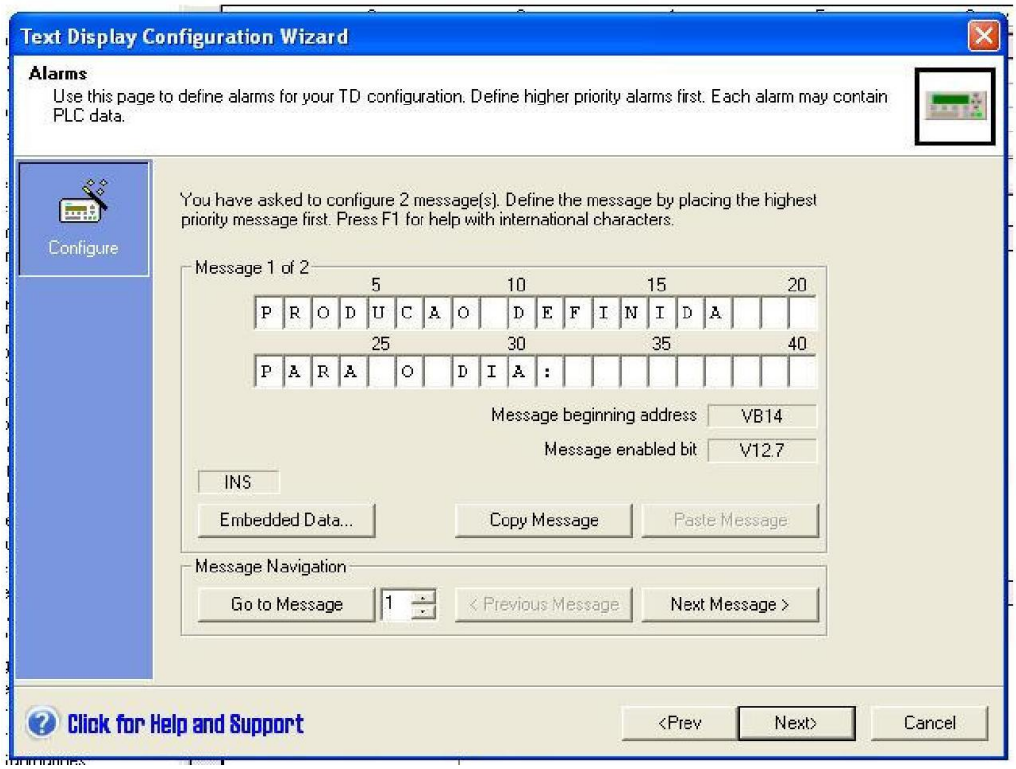
Configuração da Mensagem:


Text Display Configuration Wizard
Function Keys and Update Rate
Use this page to select options for the function keys, and select an update rate for alarms.
Configure
The TD has 8 function keys (F1 through F4 and SHIFT F1 through SHIFT F4) that are used to set memory bits in the PLC. Eight bits of memory (M bits) must be reserved for the TD to set them when a function key is pressed. One M bit is set by the TD each time the corresponding function key is pressed.
Which byte of M memory would you like to reserve for the TD function keys?
0
(0-31)
For TD 200 Version 2.1 and newer, the function keys may be configured to either set the associated M-bit, or to act as a momentary contact, setting the bit only while the function key is pressed. How should the F-keys be configured?
F-keys should set M-bits
The update rate determines how often the TD polls the PLC for messages to display. How often would you like the TD to poll for messages?
As fast as possible

(Fig. 125 – Configuração da mensagem no TD 200 (solução do exercício)


Text Display Configuration Wizard
Alarm Options
Use this page to specify the text length of alarms, and to select how they should be displayed.
Configure
The TD 200 allows two messages sizes. Please select the desired message size.
20 character message mode - displays two messages at a time
40 character message mode - displays one message at a time
The TD 200 allows you to configure up to 80 messages. How many messages do you wish to configure?
2
message(s)
Click for Help and Support
<Prev
Next>
Cancel


Text Display Configuration Wizard
Allocate Memory
Use this page to select the V-Memory block that will be used by this TD configuration
Configure
You must define where the 12 byte parameter definition will be placed in the data block. It is normally located at VB0.
Starting byte for 12 byte parameter block
0
You have defined 2 message requiring 1 byte(s) for message enable flags. You must now define where the enable flags will reside in the data block.
Starting byte for enable flags
12
You have defined 2 message requiring 80 consecutive bytes for the message information. You must define where the message information is to reside in the data block.
Starting byte for message information
14
Click for Help and Support
<Prev
Next>
Cancel


Text Display Configuration Wizard
Alarms
Use this page to define alarms for your TD configuration. Define higher priority alarms first. Each alarm may contain PLC data.
Configure
You have asked to configure 2 message(s). Define the message by placing the highest priority message first. Press F1 for help with international characters.
Message 1 of 2
5 10 15 20
PRODUCAO DEFINIDA
25 30 35 40
PARA O DIA:
Message beginning address
VB14
Message enabled bit
V12.7
INS
Embedded Data...
Copy Message
Paste Message
Message Navigation
Go to Message
1
< Previous Message
Next Message >
Click for Help and Support
<Prev
Next>
Cancel

Após escrever a mensagem PRODUÇÃO DEFINIDA PARA O DIA, precisamos que o próximo campo seja a área onde o operador possa inserir a quantidade de peças que serão produzidas. Para que isto seja possível devemos utilizar a opção Embedded Data (embutindo dados), conforme indicado na tela acima. Após clicar nesta opção surgirá a janela abaixo:

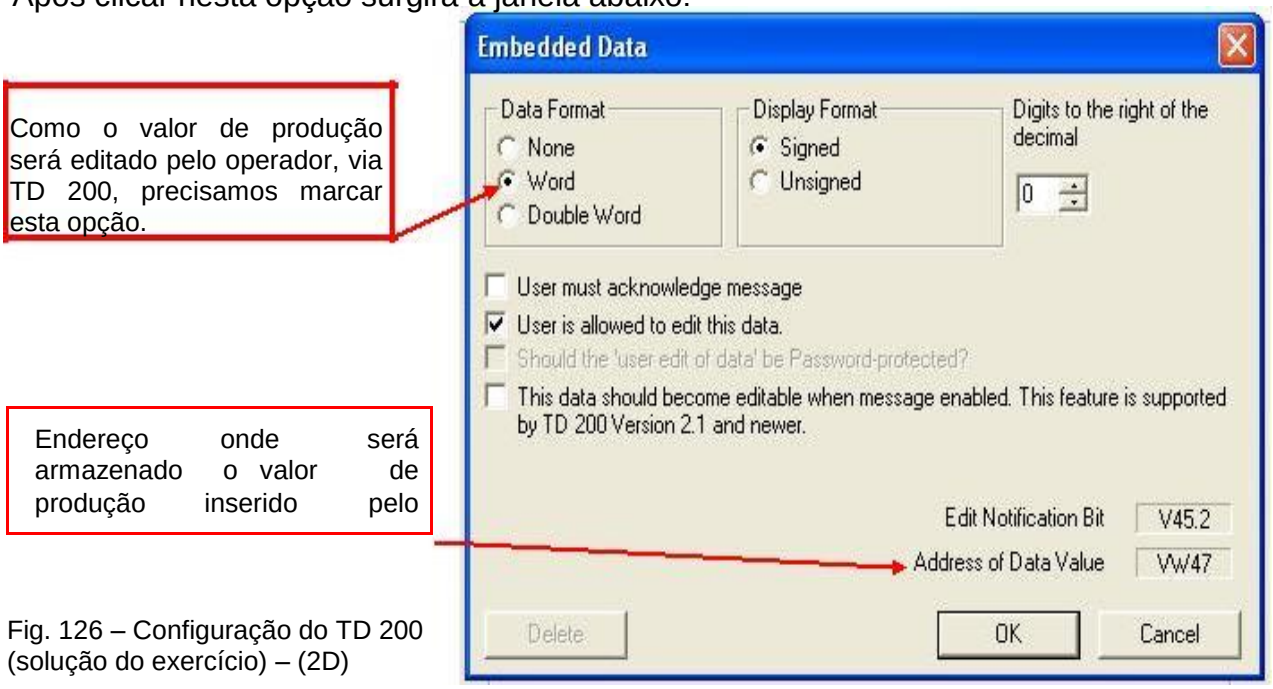

Como o valor de produção será editado pelo operador, via TD 200, precisamos marcar esta opção.
Endereço onde será armazenado o valor de produção inserido pelo
Embedded Data
Data Format
None
Word
Double Word
Display Format
Signed
Unsigned
Digits to the right of the decimal
0
User must acknowledge message
User is allowed to edit this data.
Should the 'user edit of data' be Password-protected?
This data should become editable when message enabled. This feature is supported by TD 200 Version 2.1 and newer.
Edit Notification Bit
V45.2
Address of Data Value
VW47
Delete
OK
Cancel

Fig. 126 – Configuração do TD 200 (solução do exercício) – (2D)

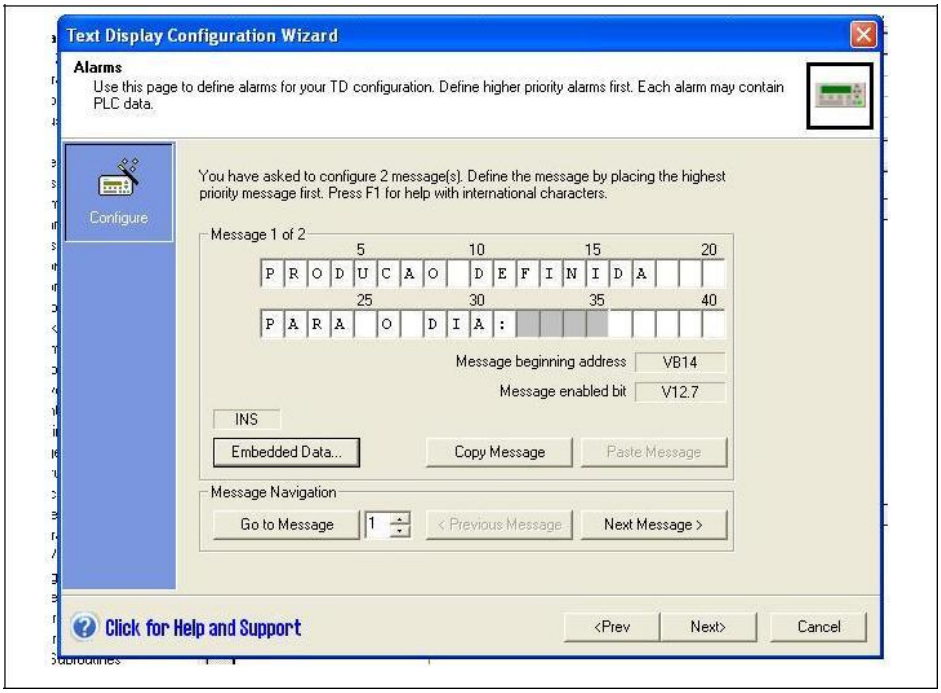

Text Display Configuration Wizard
Alarms
Use this page to define alarms for your TD configuration. Define higher priority alarms first. Each alarm may contain PLC data.
Configure
You have asked to configure 2 message(s). Define the message by placing the highest priority message first. Press F1 for help with international characters.
Message 1 of 2
PRODUCAO DEFINIDA
PARA O DIA:
Message beginning address
VB14
Message enabled bit
V12.7
INS
Embedded Data...
Copy Message
Paste Message
Message Navigation
Go to Message
Previous Message
Next Message >
Click for Help and Support
<Prev
Next>
Cancel

Fig. 127 - Configuração do TD 200 (solução do exercício) – (3D)

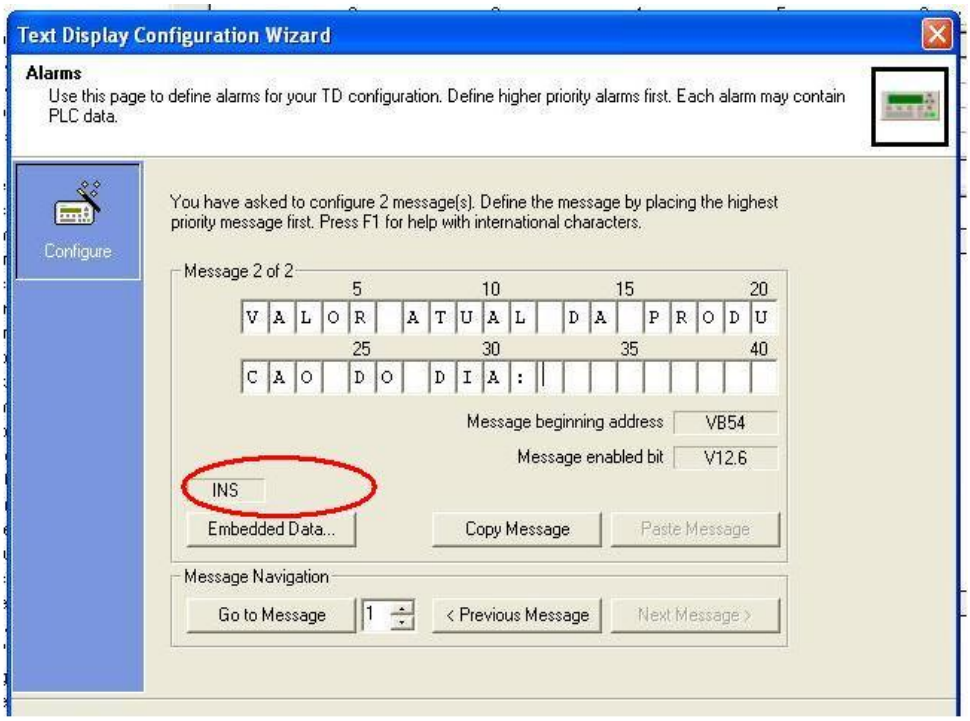

Text Display Configuration Wizard
Alarms
Use this page to define alarms for your TD configuration. Define higher priority alarms first. Each alarm may contain PLC data.
Configure
You have asked to configure 2 message(s). Define the message by placing the highest priority message first. Press F1 for help with international characters.
Message 2 of 2
5 10 15 20
VALOR ATUAL DA PRODU
25 30 35 40
CAO DO DIA:
Message beginning address VB54
Message enabled bit V12.6
INS
Embedded Data...
Copy Message
Paste Message
Message Navigation
Go to Message
1
< Previous Message
Next Message >

Fig. 128 - Configuração do TD 200 (solução do exercício) – (4D)

Após escrever a mensagem VALOR ATUAL DA PRODUÇÃO DO DIA, precisamos que no próximo campo seja mostrado o valor de contagem do contador. Para que isto seja possível devemos utilizar a opção Embedded Data (embutindo dados), conforme indicado na figura anterior. Após clicar nesta opção surgirá a janela abaixo:

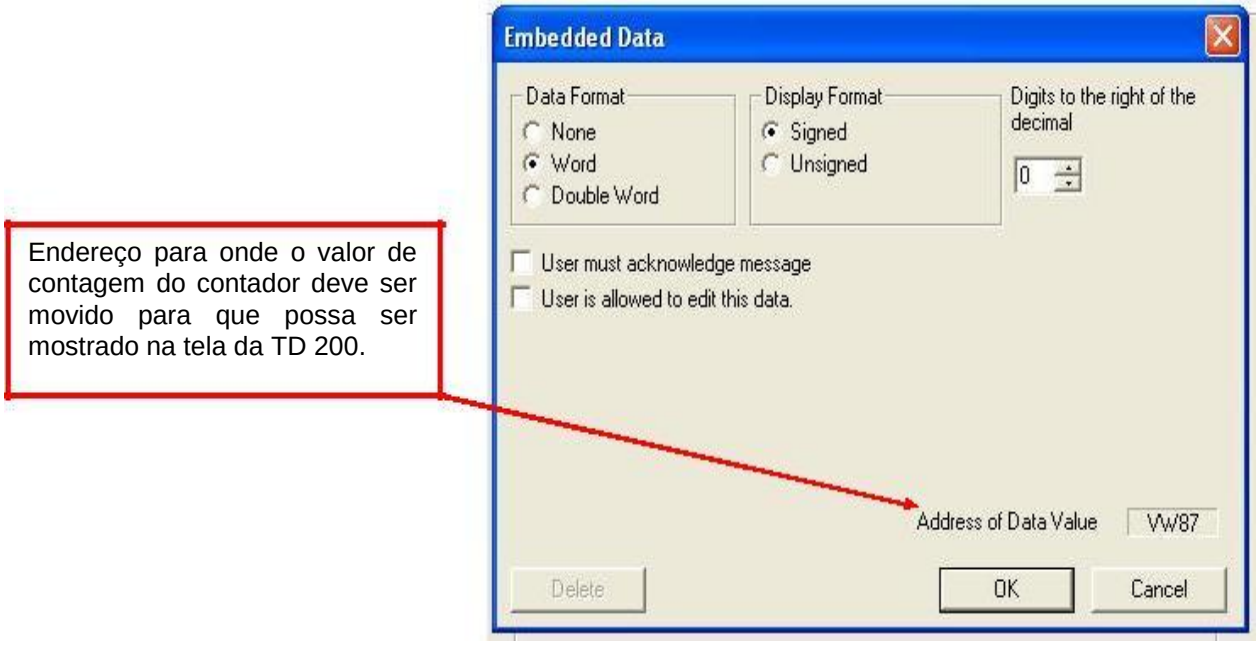
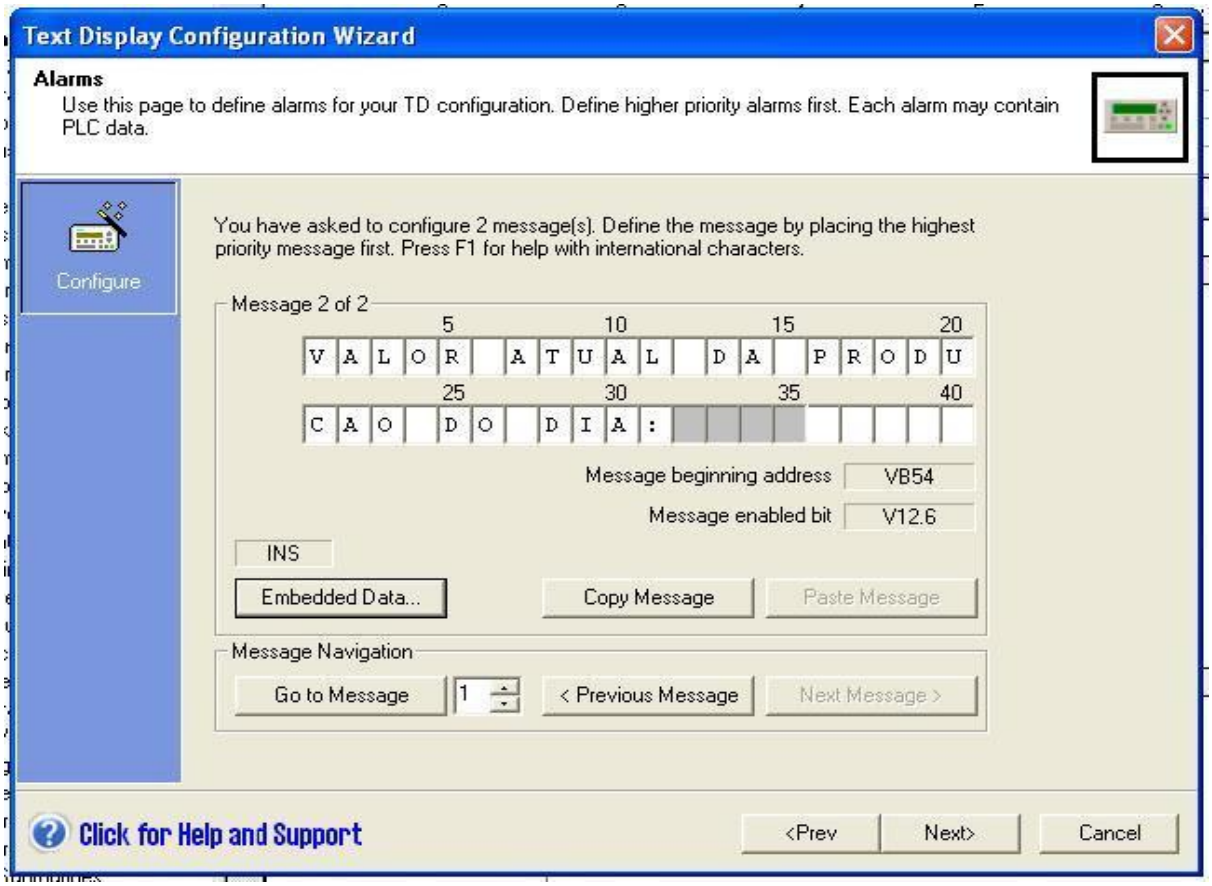


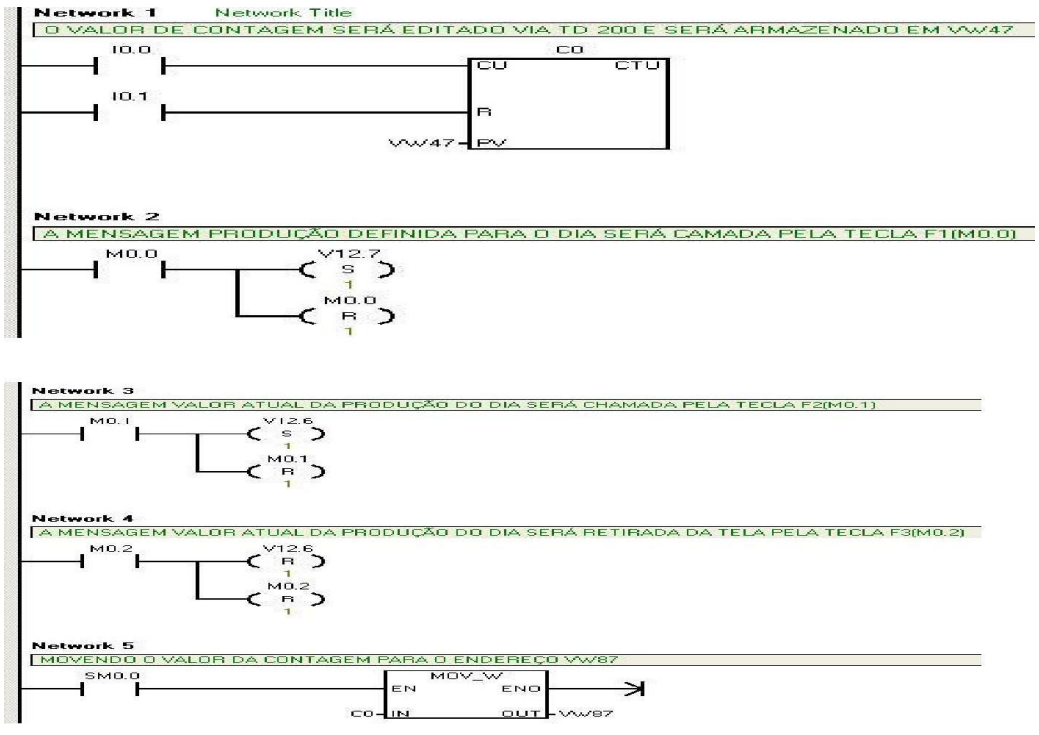
Fig. 129 - Configuração do TD 200 (solução do exercício) – (5D)

Fig. 131 – rotina de programação (solução do exercício) – (6D)


Text Display Configuration Wizard
Alarms
Use this page to define alarms for your TD configuration. Define higher priority alarms first. Each alarm may contain PLC data.
Configure
You have asked to configure 2 message(s). Define the message by placing the highest priority message first. Press F1 for help with international characters.
Message 2 of 2
5 10 15 20
VALOR ATUAL DA PRODU
25 30 35 40
CAO DO DIA:
Message beginning address VB54
Message enabled bit V12.6
INS
Embedded Data...
Copy Message
Paste Message
Message Navigation
Go to Message
1
< Previous Message
Next Message >
Click for Help and Support
<Prev
Next>
Cancel

exercício) – (7D)

Programa:



I. CONCLUINDO

Estamos concluindo este material.

A nossa expectativa é que você possa continuar aprendendo, pois novos conceitos, técnicas e tecnologias surgirão. Como dissemos no início deste material, a **automação** representa um campo no qual o avanço tecnológico é uma constante. Esse dinamismo se refletirá em dúvidas, questionamentos e novas perguntas. E é bom que seja assim, pois isto o estimulará a prosseguir em seus estudos e a aperfeiçoar seu desempenho profissional.

Discutimos a automação sob a perspectiva de um amplo cenário, constituído por conceitos, técnicas, “fazeres”, além de suas implicações e conseqüências para o setor industrial e para a vida das pessoas. Não podemos esquecer que a automação representa uma construção humana, como outras nascidas do natural desejo que as pessoas têm de conhecer, de descobrir e de buscar alternativas para facilitar a vida. Você certamente percebeu também que é a automação uma ferramenta fortíssima para o aumento da competitividade das empresas, num ambiente de acirrada concorrência, como hoje observamos.

Esperamos encontrá-lo nos próximos cursos. Até muito breve!

REFERÊNCIAS BIBLIOGRÁFICAS

CASTRO, F.C.C de. **Portas Lógicas Básicas.** Flórida, EUA, 2003 **.** Disponível em: http://pt.wikipedia.org/wiki/Dispositivo_l%C3%B3gico_program%C3%A1vel em 10/11/2008.

SILVEIRA, Paulo R. Da ; SANTOS, Winderson E. **Automação e Controle Discreto.** 7ª edição. São Paulo: Ed. Érica, 2006.

SIEMENS. **S7-200 Programmable Controller System Manual.** 8ª edição**.** Alemanha, 2008.

RABEL, Giovanni Francisco; BRAGA Jr, Roberto Alves. **Notas de Aula – Disciplinas Instrumentação (Graduação e PG). Minas Gerais.** UFLA Universidade Federal de Lavras, 2003.

SUPERTEC. Superintendência de Representações Técnicas. **Curso Micro CLP S7200,** 2002. diapositivo color.

PINTO, Joel Rocha; FERRAZ, Willerson Moreira. **Controladores Lógicos Programáveis.** São Paulo, Faculdade de Engenharia de Sorocaba e Flash Engenharia e desenvolvimento LTDA, 2007. Diapositvo color.